2 Projektorganisation

2.1 DIE ZIELDEFINITION

In der betrieblichen Softwareentwicklungspraxis besteht zu Projektbeginn haufig das Problem, dass die Projektziele unklar bleiben bzw. unzureichend definiert werden. Dies fuhrt zu mangelnder Zielorientierung und inadŠquater Losungssuche, was sich in falschen Entscheidungen und Losungen niederschlagen kann. Eine gewisse Klarheit bez. ubergeordneter Projektziele, welche in der Regel okonomischer, technischer, organisatorischer und qualifikatorischer Natur sind, sollte daher vor der Realisierung bestehen. Im Rahmen eines solchen Zielbildungsprozesses sind folgende Schritte zu durchlaufen:

(1) Zielideen generieren,

(2) Zielkatalog formulieren,

(3) Ziele gewichten,

(5) Zielbeziehungen analysieren,

(6) Zielkonflikte bereinigen,

(7) Ziele operationalisieren,

(8) Zieldefinition formulieren.

Eine schriftliche Dokumentation der Zieldefinition ist eine wesentliche Voraussetzung zur Durchsetzung der Ziele und liefert eine wichtige Grundlage fur die Gestaltung des Projektes auf operativer Ebene. Eine konsensorientierte, interessenausgleichende Zieldefinition in technisch-organisatorischer und betriebswirtschaftlicher Hinsicht zwischen Geschaftsleitung, EDV- und Fachbereich ist dabei als eine der zentralen Voraussetzungen fur den Projekterfolg anzustreben.

Damit die formulierten Ziele immer auf einem aktuellen Stand gehalten werden konnen, mŸssen im Laufe des weiteren Prozesses Kontrollen durchgefuhrt werden, da z.B. Anderungen im Geschaftsumfeld oder im technischen Umfeld Ð welche vor allem bei grossen Projekten mit entsprechender Laufzeit wahrscheinlich sind Ð die Modifikation der formulierten Ziele erfordern. Diese Aktualisierung sollte im Rahmen solcher Kontrollschritte geleistet werden.

Rauterberg, M., Spinas, P., Strohm, O., Ulich, E. & Waeber, D. (1994).
Benutzerorientierte Software-Entwicklung Konzepte, Methoden und Vorgehen zur Benutzerbeteiligung.
(Mensch - Technik - Organisation. Band 3), vdf Hochschulverlag AG an der ETH Zurich.

Ein iterativ-zyklisches Vorgehen bei der Softwareentwicklung ist auf die Änderungsdynamik von Zielen und Anforderungen ausgerichtet. Dies ist eine Notwendigkeit, damit solche Zielmodifikationen auch umgesetzt bzw. realisiert werden können.

2.2 DIE PROJEKTKLASSIFIKATION

Eine oberflächliche Analyse von Projektrahmenbedingungen bzw. eine unsystematische Projektklassifikation führt zur Abwicklung von 'Einheitsprojekten', wobei unterschiedlichen Ausgangslagen verschiedener Projekte nicht adäquat Rechnung getragen wird. Dies kann zur Folge haben, dass Ziele nicht erreicht werden oder – im schlechtesten Fall – ein Projektabbruch nötig wird. Im folgenden Abschnitt werden wesentliche Rahmenbedingungen bei der Softwareentwicklung beschrieben, die der Projektklassifikation dienen.

Für ein aufgaben- und benutzerorientiertes Projektmanagement sind folgende Rahmenbedingungen von besonderer Wichtigkeit.

2.2.1 Der Projekttyp

Vier charakteristische Projekttypen sind zu unterscheiden:

Typ A: Entwicklung einer Individuallösung für eine firmeninterne Fachabteilung.

Typ B: Entwicklung einer Individuallösung für einen externen Kunden.

Typ C: Entwicklung einer Branchenlösung für einen weitgehend bekannten Kundenkreis.

Typ D: Entwicklung einer Standardlösung für einen breiten und anonymen Benutzerkreis.

Immer häufiger werden auch Projekte abgewickelt, bei denen branchenspezifische Softwarelösungen ausgewählt und durch die interne Entwicklungsabteilung oder eine externe Softwarefirma bzw. den Softwarelieferanten firmenspezifisch angepasst werden.

Die verschiedenen Projekttypen unterscheiden sich erheblich in den Möglichkeiten zur Aufgaben- und Benutzerorientierung bei der Softwareentwicklung, da von Projekttyp A zu D

(1) die Bekanntheit der Benutzer,

(2) die Bekanntheit der zu unterstützenden Aufgaben,

(3) die Bekanntheit des organisatorischen Kontextes und

(4) die Zugangsmöglichkeiten zu den Benutzern

immer weniger gegeben sind. Die raumzeitliche Nähe zwischen Benutzern und Softwareentwicklern wie auch die infrastrukturellen Voraussetzungen für deren Zusammenarbeit sind z.B. in Abhängigkeit vom Projekttyp sehr unterschiedlich. Entsprechend sind verschiedene Formen der Projektorganisation und Zusammenarbeit zu praktizieren. Unterschiedliche Varianten der Projektorganisation werden daher in Teil B, Kapitel 2.3 beschrieben.

2.2.2 Die Projektkomplexität

Die Komplexität von Softwareprojekten wird u.a. durch folgende Merkmale bestimmt:

- Umfang der geplanten Systemfunktionalität,
- Anzahl und Unterschiedlichkeit der betroffenen Fachaufgaben (siehe dazu auch Benutzer- und Aufgabenkontext),
- Umfang der geplanten arbeitsorganisatorischen Veränderungen für die Fachbereiche bzw. Anwendungsbereiche,
- Umfang der geplanten qualifikatorischen Massnahmen für die potentiellen Systembenutzer.

Anhand dieser Merkmale eingestuft, ist etwa die Entwicklung eines Lieferantenverwaltungssystems in einer Einkaufsabteilung als ein Projekt von eher geringer Komplexität zu betrachten. Die Entwicklung eines integrierten Produktionsplanungs- und -steuerungssystems, das sowohl die informationstechnische wie auch die organisatorische Integration der Bereiche Verkauf, Disposition, Einkauf und Fertigung zum Gegenstand hat, zeichnet sich dagegen durch eine sehr hohe Komplexität aus. Komplexe Projekte sind in der Regel durch eine lange Projektdauer, einen grossen Personenaufwand sowie eine grosse Anzahl von Projektbeteiligten gekennzeichnet.

2.2.3 Die Projektinnovativität

Eine wesentliche Rahmenbedingung ist die Innovativität bzw. Neuartigkeit eines Projektes. Innovative Projekte sind – im Vergleich zu Routineprojekten – durch eines oder mehrere der folgenden Merkmale gekennzeichnet:

(1) Das System ist eine Neu- oder Erstentwicklung,

(2) in der Entwicklungsabteilung bestehen keine Erfahrungen mit diesem Fachbereich, dieser Branche bzw. der Art der zu unterstützenden Aufgaben,

(3) in der Entwicklungsabteilung bestehen keine Erfahrungen mit dieser Art von Anwendungen, Automatisierung, Techniklösung,

(4) im Fach- bzw. Anwendungsbereich besteht eine geringe EDV-Unterstützung und -Erfahrung.

Bei innovativen Projekten ist also das Zurückgreifen auf Erfahrungswerte nur bedingt möglich. Die Planung solcher Projekte ist deshalb erschwert. Vor diesem Hintergrund setzen innovative Projekte umfassende Informationen über den Fachbereich (benötigte Daten und Funktionen, Arbeitsabläufe usw.) voraus. Umfassende Analysen sowie konzeptionelle Arbeiten mit den Benutzern zusammen (zur Nutzung ihres Fachwissens!) bilden eine wesentliche Voraussetzung für den Erfolg von innovativen Softwareprojekten.

2.2.4 Der Benutzer- und Aufgabenkontext

Anwendungssoftware wird zur Unterstützung von sehr unterschiedlichen Benutzergruppen und Aufgabentypen entwickelt. Benutzer- und aufgabenorientierte Softwareentwicklung erfordert daher, dass dieser Unterschiedlichkeit beim Vorgehen Rechnung getragen wird. Die Unterschiedlichkeit von Benutzergruppen zeigt sich vor allem an folgenden Merkmalen:

(1) Anzahl der potentiellen Benutzer,

(2) fachlicher bzw. qualifikatorischer Hintergrund der Benutzer,

(3) hierarchische Ebene der Benutzer,

(4) EDV-Erfahrung der Benutzer.

Aufgaben im Büro- und Verwaltungsbereich unterscheiden sich u.a. in den folgenden Merkmalen:

(1) Strukturiertheit, Formalisierbarkeit der Aufgaben,

(2) Vielfalt und Variabilität der (Teil-)Aufgaben,

(3) Kommunikations- und Kooperationserfordernisse,

(4) Umfang der Freiheitsgrade im Aufgabenvollzug,

(5) Struktur und Menge erforderlicher Informationen, Funktionen,

(6) Stellenwert der EDV bei der Aufgabenerfüllung.

Kasten 15: Projekt Kundenverwaltung.

Die interne Entwicklungsabteilung einer Kleinbank ist damit beauftragt, ein neues System zur Kundenverwaltung zu entwickeln. Das alte funktionstastengesteuerte System, das auf Stand-alone-PCs implementiert ist, soll durch ein System mit Window-Oberfläche und leicht erweiterter Funktionalität ersetzt werden; zudem sollen die Stand-alone-PCs miteinander vernetzt werden. Die Arbeitsorganisation der betroffenen Fachabteilung – in der fünf Teilzeitmitarbeiterinnen mit der Erfassung und Pflege von Kundenstammdaten beauftragt sind – soll im Rahmen des Projektes weitgehend unverändert bleiben.

Kasten 16: Projekt Bibliothek.

Eine Softwarefirma ist damit beauftragt, ein integriertes Bibliothekssystem für eine Universität zu entwickeln. Der Bibliotheksbetrieb wurde bisher weitgehend konventionell mit Mikrofichen, Katalogen, Bestellformularen usw. abgewickelt. Mit Hilfe eines interaktiven On-line-Systems sollen nun eine komfortable Anlage und Pflege von Bibliotheksdatensätzen, komplexe Abfragemöglichkeiten, die Vorbestellung von Büchern usw. universitätsintern und dezentral in externen Instituten mittels eines Netzwerkes ermöglicht werden.

Potentielle Systembenutzer sind das Bibliothekspersonal, Sekretärinnen, wissenschaftliche Mitarbeiter, Dozenten, Studierende usw. Das Bibliothekspersonal setzt sich aus gelernten Bibliothekaren, Sachbearbeiterinnen sowie studentischen Hilfskräften zusammen. Die bisher praktizierte Arbeitsteilung in der Bibliothek soll bei der Systemkonzeption neu überdacht werden. Es ist vorgesehen, das System auf dem universitätsinternen Grossrechner zu implementieren. Für das wissenschaftliche Personal soll dabei die Möglichkeit bestehen, mittels den an ihren Arbeitsplätzen installierten PCs Zugriff auf das zukünftige Bibliothekssystem zu haben.

Der Stellenwert des EDV-Systems bei der Aufgabenerfüllung zeigt sich u.a. daran, ob ein System als integrales, zentrales Hilfsmittel für den Grossteil der Aufgaben oder nur zur Unterstützung einzelner Aufgaben benötigt wird.

Aus den beschriebenen Rahmenbedingungen wird deutlich, dass die Ausgangslage bei Softwareprojekten sehr unterschiedlich sein kann. An zwei Projektvorhaben (Projekt Kundenverwaltung und Projekt Bibliothek) wird dies veranschaulicht (vgl. Kasten 15 und Kasten 16). Entsprechend den Projektbeschreibungen lassen sich die in Kasten 15 und 16 dargestellten Projekte wie folgt klassifizieren:

Rahmenbedingungen	Projekt Kundenverwaltung	Projekt Bibliothek
Projekttyp	A	B
Projektkomplexität	gering	hoch
Projektinnovativität	gering	hoch
Benutzergruppen	homogen	heterogen
Aufgaben	einfach, Routine	vielfältig, anspruchsvoll

Grundsätzlich ist bei der Zieldefinition und Projektklassifikation zu beachten, dass diese Projektschritte in einem Wechselverhältnis stehen (Massnahmen der Aufgabenerweiterung als arbeitsorganisatorische Projektziele erhöhen z.B. auch die Vielfältigkeit und Variabilität der betroffenen Fachaufgaben). Daher sollten diese Projektschritte aufeinander abgestimmt und unter Zusammenarbeit der Auftraggeber und Auftragnehmer sowie der Projektbeteiligten aus den Fach- und EDV-Bereichen erfolgen.

2.3 DIE AUFBAUORGANISATION

Die Aufbauorganisation regelt u.a. die Informations- und Entscheidungswege, Verantwortlichkeiten sowie Arbeitsteilung in einem Projekt. Dabei werden häufig strategische, steuernde sowie operative Gremien bzw. Funktionsträger unterschieden. Durch die Aufbauorganisation ist u.a. festgelegt, welche Personen bzw. Funktionsträger welchem Gremium angehören und welche Aufträge bzw. Aufgaben im Verantwortungsbereich der einzelnen Gremien liegen. Wesentliche Kriterien für die Bewertung der Aufbauorganisation sind u.a. die Zentralität vs. Dezentralität, die Möglich-

keiten zur Mitwirkung der Projektbeteiligten und -betroffenen sowie das Strukturierungsprinzip im Sinne funktionaler Arbeitsteilung vs. funktionale Integration (vgl. Ulich 1994).

2.3.1 Die traditionelle Aufbauorganisation

Traditionelle Modelle der Aufbauorganisation von Softwareprojekten entsprechen der in Abbildung 5 dargestellten Struktur. Aus der Darstellung in Abbildung 5 lassen sich eine Reihe von Hauptmerkmalen und Problemen traditioneller Strukturen bei der Softwareentwicklung ableiten.

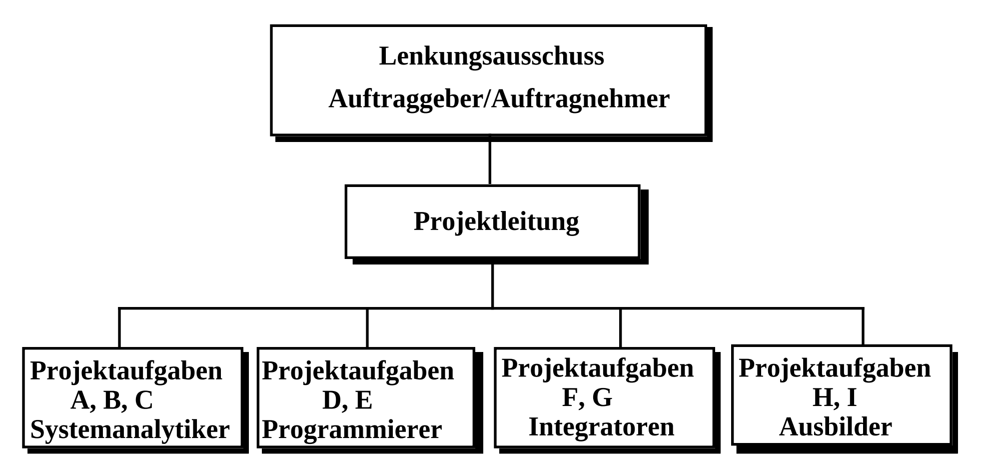


Abbildung 5: Vereinfachte Darstellung eines traditionellen Modells einer Aufbauorganisation.

1. *Die Zielvorgabe, Terminierung und Budgetierung erfolgt zentral, z.B. vom Lenkungsausschuss, das heisst, ohne Einflussmöglichkeiten der am Entwicklungsprozess direkt Beteiligten und Projektmitarbeiter auf operativer Ebene.*

Damit kann ein grosser Kosten- und vor allem auch Termindruck für die Projektmitarbeiter auf operativer Ebene verbunden sein. Starker Termindruck führt häufig zu Kreativitätshemmnissen, Überbeanspruchung und sogar Überforderung bei den Projektmitarbeitern sowie Leistungs- und Qualitätseinbussen beim Produkt. Für die Projektbeteiligten auf operativer Ebene kann diese Situation zudem auch frustrierend sein, wenn z.B. das

Management – von strategischen Vorgaben abgesehen – wenig Engagement und Interesse für das Projekt auf operativer Ebene zeigt. Deshalb ist nach aufbauorganisatorischen Strukturen zu suchen, welche den Projektmitarbeitern erlauben, sich an solchen Entscheidungen zu beteiligen, und Spielräume offenlassen. Die Projektmitarbeiter sollten in diesem Sinne Einflussmöglichkeiten auf folgende Entscheidungen haben:

- Planung und Festlegung von Teilzielen,
- Planung und Festlegung des Projektmanagements auf operativer Ebene,
- Budgetierung und Terminierung von Teilaufgaben.

2. *Eine repräsentative und aktive Mitarbeit der Benutzer ist nicht vorgesehen.*

Eine organisatorische Integration der Benutzer in den Lenkungsausschuss, in Projektgruppen und/oder in Arbeitsgruppen als Voraussetzung für eine repräsentative und aktive Mitarbeit, bei der Benutzervertreter

- an den verschiedenen Projektphasen beteiligt sind,
- zu wesentlichen Planungs- und Gestaltungsfragen beigezogen werden,
- Systemvorschläge entwickeln und auch
- Mitentscheidungsmöglichkeiten haben,

ist nicht vorgesehen und auch nicht möglich. Mögliche Folgen davon sind Softwarelösungen, die an den Bedürfnissen der Benutzer vorbeizielen und zu Unzufriedenheit bei den Benutzern führen. Traditionelle Modelle der Aufbauorganisation bei der Softwareentwicklung sind daher u.a. vor dem Hintergrund einer unzureichenden Benutzerbeteiligung abzulehnen.

3. *Die Projektorganisation ist funktionsorientiert und stark arbeitsteilig.*

Traditionell werden vor allem grosse Projekte – die in entsprechend grossen Softwarefirmen und/oder grossen firmeninternen Entwicklungsabteilungen realisiert werden – funktional in Bereiche wie z.B. Systemanalyse, -konzept, Programmierung, Integration, Ausbildung usw. gegliedert. Bürokratie, lange Dienstwege sowie eine grosse Arbeitsteilung und Spezialisierung sind kennzeichnend für solche Strukturen. Ein entsprechender Aufwand für Administration und Koordination sowie eingeschränkte Flexibilität sind einige Folgen dieser Form der Projektorganisation.

In diesem Zusammenhang ist das aus dem soziotechnischen Systemkonzept abgeleitete Prinzip der *'Schaffung von relativ unabhängigen Organisations- bzw. Projekteinheiten'*, in denen Schwankungen und Störungen selbst reguliert werden können, von besonderer Bedeutung. Dieses Prinzip impliziert, Projektgruppen nicht nach funktionalen Gesichtspunkten, sondern produkt- bzw. modulorientiert zu bilden. Dementsprechend sollten grosse Projekte in abgeschlossene Teilprojekte gegliedert werden, damit vollständige Aufgabenstellungen – die sowohl analytische, konzeptionelle, spezifizierende, realisierende und kontrollierende Teiltätigkeiten umfassen – verschiedenen Projektgruppen zur Bearbeitung übertragen werden können. Damit werden die Projektgruppen mit vollständigen und somit motivierenden Aufgaben beauftragt, die u.a. das selbständige Setzen von Zielen, die Wahrnehmung von Planungsfunktionen, die Auswahl von Arbeitsmitteln, Ausführungsfunktionen sowie Kontrollfunktionen mit Resultatfeedback umfassen (vgl. Volpert 1987).

2.3.2 Alternative Modelle der Aufbauorganisation

Alternative Modelle der Aufbauorganisation sind in traditionellen Projektmanagementmodellen meist nicht vorgesehen. Die dargestellten Problembereiche lassen allerdings Zweifel aufkommen, ob in traditionellen Projektstrukturen in effizienter Weise aufgaben- und benutzungsangemessene Software entstehen kann.

In diesem Abschnitt werden deshalb drei Organisationsmodelle beschrieben, die den dargestellten Problemen entgegenwirken und die Kooperation zwischen Softwareentwicklern und Benutzern unterstützen (siehe dazu auch Kubicek 1979).

In allen Modellen werden die Instanzen 'Lenkungsausschuss', 'Projektleitung' und 'Projektgruppe' unterschieden. Diese Instanzen haben generell die im folgenden angegebenen Aufgaben.

Lenkungsausschuss

Grundsätzlich übernimmt der Lenkungsausschuss als strategisches Organ die globale Steuerung, Überwachung und Kontrolle des Projektes. Eine interessenausgleichende Zielvorgabe, das Bereitstellen finanzieller, personeller und zeitlicher Ressourcen, das Setzen von Prioritäten sowie die Unterstützung des Projektleiters und der Projektgruppen auf operativer Ebene gehören zu diesem Aufgabenspektrum. Dies beinhaltet auch, dass der Lenkungsausschuss z.B. Aufträge für die Projektgruppe vorbereitet, Aufträge

freigibt, Anfragen der Projektgruppe beantwortet und zu deren Arbeitsergebnissen Stellung nimmt.

Projektleitung

Die Projektleitung hat für die projektzielgerechte Entwicklung des Systems zu sorgen. Daher steuert und koordiniert der Projektleiter die Aktivitäten zwischen Lenkungsausschuss, Projekt- und Arbeitsgruppen. Die Funktion der Projektleitung zielt – im Unterschied zur traditionellen Rolle der Projektleitung – u.a. darauf ab, Rahmenbedingungen zu schaffen, die eine motivierte und effiziente Projektarbeit auf operativer Ebene fördern (siehe dazu ausführlich Teil B, Kapitel 2.3.3).

Projektgruppe

Die Projektgruppe plant, realisiert und kontrolliert das Projekt auf operativer Ebene. Dies umfasst neben organisatorischen Aufgaben v.a. auch inhaltliche Aufgaben wie Systemanalyse, Anforderungsermittlung, Systemkonzeption, Systemspezifikation, Implementation, Systemtests, Systemeinführung sowie die begleitende Qualitätssicherung (siehe dazu bei Wallmüller 1990; siehe auch Teil B, Kapitel 2.3.4).

Die im folgenden beschriebenen Modelle zeigen unterschiedliche Möglichkeiten der Aufbauorganisation sowie Varianten der Zusammensetzung von Projektgremien. In Abhängigkeit von Projektzielen, dem Projekttyp und weiteren Rahmenbedingungen können geeignete Modelle für konkrete Vorhaben ausgewählt und bez. der konkreten Aufgaben und Kompetenzen der einzelnen Instanzen sowie der Informationsflüsse zwischen den Instanzen spezifiziert werden.

Da heutzutage meist verschiedene Fachbereiche in einem Unternehmen von der Entwicklung neuer Anwendungen oder der Weiterentwicklung bestehender Software betroffen sind, wird in den folgenden Modellen der *organisatorischen Integration von Fachbereichen* besondere Bedeutung beigemessen. So können z.B. bei der Entwicklung und Einführung eines integrierten Bürokommunikationssystems in einem Handelsunternehmen die aufgaben- und mitarbeiterspezifisch unterschiedlichen Interessen nur durch den Einbezug der Bereiche Verkauf, Kalkulation, Rechnungswesen, administrative Auftragsabwicklung, Einkauf, Lager und Versand in das Projekt angemessen berücksichtigt werden (zur repräsentativen Vertretung von Anwendern, Benutzern siehe ausführlich Teil A, Kapitel 2.3.1).

Das Modell der gemischten Projektgruppen

Beim Modell der gemischten Projektgruppen setzt sich die Projektgruppe aus internen und/oder externen Spezialisten wie z.B. Softwareentwicklern, Arbeitswissenschaftlern usw. und Vertretern der verschiedenen Fachbereiche zusammen (vgl. Abbildung 6).

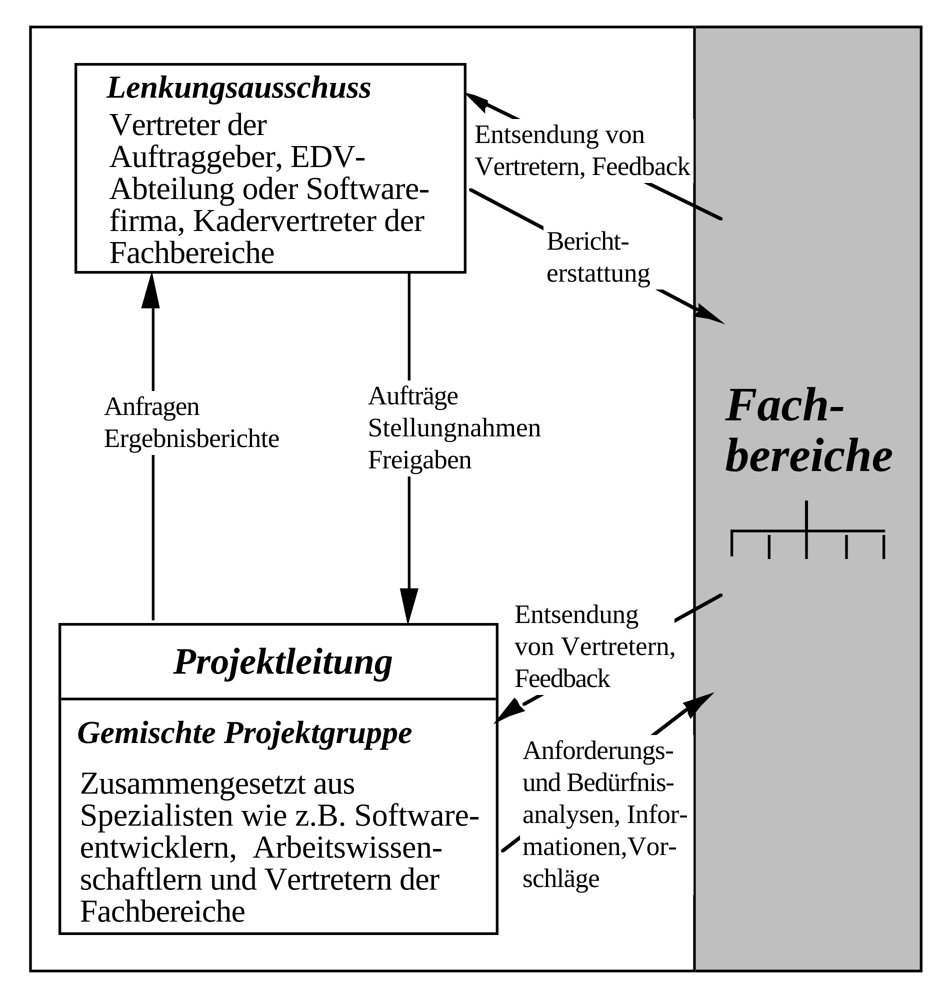


Abbildung 6: Das Modell der gemischten Projektgruppe.

Die Aufgabe der Projektgruppe ist die Konzeption und Realisierung des Systems. Die Anforderungen und Bedürfnisse der verschiedenen Fachbereiche werden im Rahmen von Anforderungs- und Bedürfnisanalysen durch die Projektgruppe erhoben. Die Fachbereiche erhalten von der Projektgruppe Informationen und Systemvorschläge. Die Fachbereiche wiederum geben – z.B. im Rahmen von Informationsveranstaltungen über das Projekt – der Projektgruppe Feedback (vgl. Abbildung 6).

Mögliche Aufgaben der Vertreter aus den Fachbereichen sind – gemeinsam mit den Softwareentwicklern und/oder Arbeitswissenschaftlern –, das bestehende technisch-organisatorische System zu analysieren, Anforderungen für das neue System bez. Funktionalität, Datenstrukturen usw. zu formulieren sowie die Entwicklung möglicher Systemlösungen, welche den Fachbereichen zur Beurteilung vorgeschlagen werden können. Um diese Aufgaben auch erfüllen zu können, müssen die Fachbereichsvertreter qualifiziert und von ihren täglichen Aufgaben zumindest zum Teil freigestellt werden (vgl. Teil A, Kapitel 2.3.1).

Die Arbeit der Projektgruppe unterliegt der Überwachung, Steuerung und Kontrolle des Lenkungsausschusses, in welchem z.B. Vertreter des Auftraggebers, Vertreter der internen EDV-Abteilung oder externen Softwarefirma sowie Kadervertreter der Fachbereiche wirken. Die Vertreter der Fachbereiche im Lenkungsausschuss bringen Anregungen, Anforderungen usw. ihres Fachbereiches im Lenkungsausschuss ein, wie sie auch umgekehrt die Fachbereiche über die Aktivitäten im Projekt informieren.

Der Vorteil dieses Modells besteht darin, dass Fachbereichsvertreter aktiv an der Planung und Realisierung des Projektes mitarbeiten, Anforderungen und Anregungen einbringen und dabei auch die verschiedenen Fachbereiche direkt über die laufenden Arbeiten auf strategischer und operativer Ebene informiert werden können. Bei Projekten mit langer Dauer besteht bei dieser Form des Benutzereinbezugs allerdings auch die Gefahr, dass die Benutzervertreter in der gemischten Projektgruppe im Laufe des Projektes den Kontakt zu ihrem Fachbereich verlieren und sich zu 'umgeschulten Systemspezialisten' entwickeln. Insofern bedingt diese Form der Projektorganisation, dass der Austausch zwischen Benutzervertretern und Fachbereichen z.B. im Rahmen von Informationsveranstaltungen, Workshops institutionalisiert wird bzw. kontinuierlich stattfindet.

Das Modell paralleler Arbeitsgruppen

Beim Modell paralleler Arbeitsgruppen wird ebenfalls von einer gemischten Projektgruppe ausgegangen. Darüber hinaus werden zusätzlich Arbeitsgruppen eingerichtet, die entweder interessen- oder aufgabenspezifisch zusammengesetzt sind.

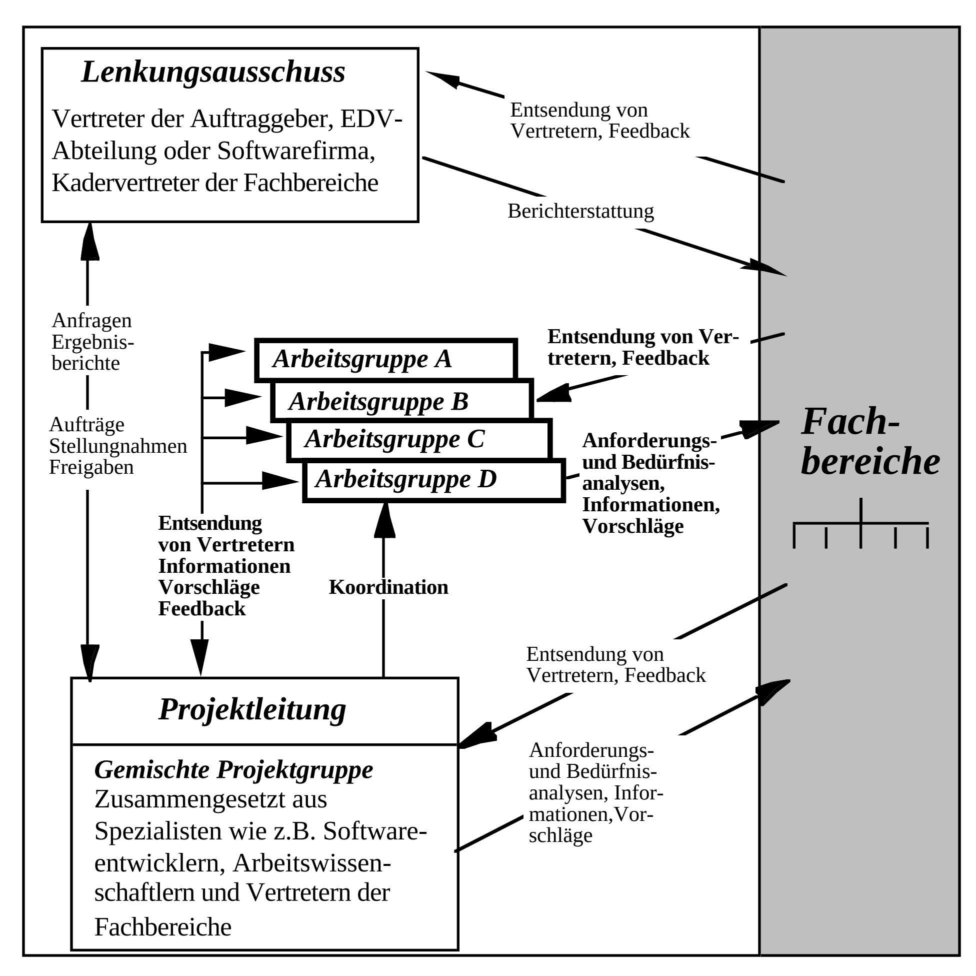


Abbildung 7: Das Modell paralleler Arbeitsgruppen.

Eine *interessenspezifische Arbeitsgruppe* setzt sich aus Vertretern verschiedener Fachbereiche – z.B. nur Abteilungsleitern – zusammen. Hierbei kann einem eventuell notwendigen Willensbildungs- und Abstimmungsprozess der einzelnen Fachbereiche auf taktischer Ebene Rechnung getragen werden.

Eine *aufgabenspezifische Arbeitsgruppe* setzt sich aus Vertretern eines oder mehrerer Fachbereiche zusammen und ist mit der Bearbeitung spezifischer Aufgaben wie z.B. der Entwicklung von möglichen Varianten der Ablauforganisation, des Bildschirm-Layouts, der Formulargestaltung usw. beauftragt (vgl. Abbildung 7).

Die einzelnen Arbeitsgruppen sollten die Grösse von ca. acht bis zehn Personen nicht übersteigen. Erfahrungsgemäss zeigt sich, dass in Arbeits- oder Projektgruppen mit dieser Grösse u.a. die Transparenz des 'sozialen Systems' erhalten bleibt und die Gefahr, dass verschiedene Subgruppen entstehen, reduziert wird. Mit dieser Grösse können gute Synergien entstehen, indem verschiedene Informationen und Wissensbestände genutzt werden.

Ein oder zwei Vertreter der Arbeitsgruppen wirken in der Projektgruppe mit und nehmen zum Austausch von Informationen, Vorschlägen und Feedback die Aufgabe von Verbindungsgliedern zwischen ihrer Arbeitsgruppe und der Projektgruppe wahr. Allgemeine Anforderungs- und Bedürfnisanalysen bei der Gesamtheit der Projektbetroffenen werden entweder durch die Projektgruppe oder durch die Arbeitsgruppen durchgeführt.

Dieses Modell ist vor allem für komplexe Projektvorhaben geeignet, bei denen umfangreiche Veränderungen des technisch-organisatorischen Systems in den Fachbereichen vorgesehen sind. Das Modell paralleler Arbeitsgruppen ermöglicht für solche Vorhaben eine 'modulare' Arbeit unter intensiver Mitarbeit der Fachbereiche. Dies setzt allerdings voraus, dass den Arbeitsgruppen in sich geschlossene Aufträge zur Bearbeitung übertragen werden.

Die adäquate Koordination der Arbeitsgruppen, welche bei diesem Modell durch die Projektleitung erfolgt, ist eine wichtige Voraussetzung für das Funktionieren dieser Form der Projektarbeit. Der Aufwand für diese Aufgabe ist entsprechend gross und ist in Projekten mit mehreren parallelen Arbeitsgruppen sinnvollerweise durch eine Assistenz des Projektleiters bzw. einen Projektkoordinator zu erfüllen.

Das Modell koordinierter Arbeitsgruppen

Eine weitgehend autonome Projektarbeit durch die Fachbereiche ist im Modell koordinierter Arbeitsgruppen vorgesehen. Dabei werden vor allem aufgabenspezifische Arbeitsgruppen eingerichtet, die sich aus Vertretern der Fachbereiche zusammensetzen. Der Projektleiter bzw. Projektkoordinator steuert und koordiniert die Arbeitsgruppen (vgl. Abbildung 8).

Hinter diesem Modell steht die Idee, dass in den Arbeitsgruppen die Analyse, Konzeptions- und Implementierungsaufgaben eines Projektes selbständig erfüllt werden bzw. die Verantwortung für deren Erfüllung bei den Arbeitsgruppen liegt. Programmiersprachen der vierten Generation bieten durchaus die Möglichkeit, dass Benutzer einfachere Applikationen selbständig realisieren. Für Aufgaben, die die Möglichkeiten der Arbeitsgruppen übersteigen, ist bei diesem Modell vorgesehen, dass Spezialisten (z.B. Softwareentwickler, Anwendungsspezialisten, Arbeitswissenschaftler) die Arbeitsgruppen beraten und in Abstimmung mit dem Lenkungsausschuss Aufträge der Arbeitsgruppen (z.B. Prototypenentwicklung, Programmierung) übernehmen.

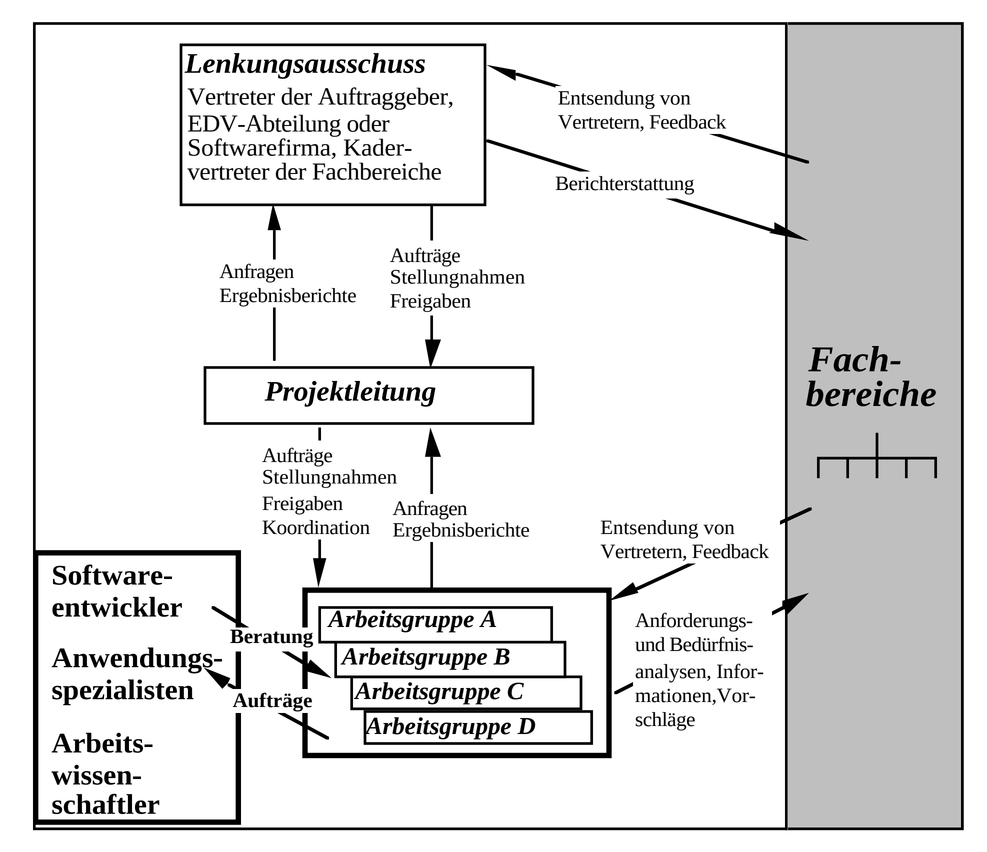


Abbildung 8: Das Modell koordinierter Arbeitsgruppen.

Voraussetzung für den Erfolg solcher koordinierter Arbeitsgruppen sind eine

(1) weitreichende fachliche und EDV-bezogene Qualifikation der Fachbereichsvertreter,

(2) ihre zeitliche Freistellung,

(3) eine modulare Auftragszuteilung in die Arbeitsgruppen sowie

(4) eine adäquate Steuerung und Koordination der Arbeitgruppen durch die Projektleitung.

Zusammenfassung

Die dargestellten Modelle unterscheiden sich deutlich in der organisatorischen Integration und den daraus resultierenden Möglichkeiten des Einflusses der Fachbereiche auf die Systementwicklung. Die Entscheidung für eines der Modelle und dessen projektspezifische Anpassung setzen die Identifikation der relevanten Interessengruppen, eine genaue Zieldefinition sowie eine Projektklassifikation auf der Basis der Rahmenbedingungen des Projektes voraus.

Bei Spezialprojekten wie z.B. der Entwicklung eines Expertensystems oder Projekten des Typs D empfiehlt sich ein Organisationsmodell, das sich weitgehend auf den Einsatz von Spezialisten stützt.

Bei 'einfacheren' Projektvorhaben des Typs A, B und C, die sich durch geringe Komplexität und Innovativität sowie einen homogenen Benutzer- und Aufgabenkontext auszeichnen, wäre z.B. das Modell 'Gemischte Projektgruppe' sinnvoll. Projekte mit hoher Komplexität und Innovativität sowie heterogenem Benutzerkreis und Aufgabenkontext erfordern dagegen die Realisierung einer auf parallelen oder koordinierten Arbeitsgruppen gestützten Aufbauorganisation.

Die erfolgreiche Praktizierung der beiden letzten Modelle setzt voraus, dass folgende Bedingungen erfüllt sind:

(1) Den Fachbereichsvertretern werden angemessene zeitliche Freiräume für die Projektarbeit gewährt.

(2) Die Fachbereichsvertreter verfügen über die notwendige EDV-Qualifikation.

(3) Die Fachbereichsvertreter besitzen grosses fachliches Anwendungswissen.

Zu den qualifikatorischen Voraussetzungen aktiver Mitarbeit von Fachbereichsvertretern wurde ausführlich im Teil A, Kapitel 2 eingegangen.

2.3.3 Die Projektleitung

Die zentrale Bedeutung des Projektleiters für eine erfolgreiche Abwicklung von Projekten wird immer wieder betont. Eine lehrbuchmässige Definition des Auftrages eines Projektleiters lautet wie folgt: "Der Projektleiter hat dafür zu sorgen, dass das spezifizierte Projektziel im gegebenen Kosten- und Terminrahmen erreicht wird." Diese Definition umfasst u.a. die Planung und Steuerung des Projektablaufes und Mitteleinsatzes, die Steuerung und

Koordination der Projektgruppe(n) auf operativer Ebene sowie die Ergebniskontrolle. Traditionell kommt dem Projektleiter dabei eine zentrale Rolle zu.

Im Unterschied zu dieser traditionellen Rolle wird dem Projektleiter gemäss den Prinzipien des soziotechnischen Systemkonzeptes eine anders akzentuierte Rolle zugewiesen. Der Projektleiter hat hierbei die Hauptaufgabe, die *Grenzbedingungen* im Projekt *zu regulieren*.

Wie bereits besprochen beinhaltet dies, dass die Aktivitäten des Projektleiters darauf abzielen, Bedingungen zu schaffen, die eine effiziente und motivierte Projektarbeit fördern. Zur Unterstützung der Arbeit der Projektgruppe nimmt der Projektleiter dabei eine 'Vermittler- und Moderatorenrolle' ein, die interne Regulation erfolgt jedoch durch die Projektgruppe selbst.

Im Zusammenhang mit dieser Moderatorenrolle bzw. diesem delegativen Führungsstil hat der Projektleiter folgende Aufgaben zu erfüllen:

(1) Zielvereinbarung zusammen mit der Projektgruppe,

(2) gemeinsame Überprüfung der Teilaufgaben nach Zielvereinbarung,

(3) Bereitstellung und Kontrolle der notwendigen Ressourcen,

(4) Bereitstellung der relevanten Informationen für die Projektarbeit,

(5) Bereitstellen von Qualifizierungsmöglichkeiten für die Projektarbeit,

(6) gemeinsame Evaluation der Ergebnisse.

Diese Rollenzuschreibung hat den Vorteil, dass der Projektleiter von operativen Aufgaben entlastet ist und die notwendigen Rahmenbedingungen für eine effiziente Aufgabenausführung in einer selbstregulierenden Projektgruppe (vgl. Teil B, Kapitel 2.3.4) schaffen kann.

Kasten 17: Ein Beispiel für die grenzregulierende Funktion des Projektleiters.

Der Projektleiter bespricht im Auftrag des Lenkungsausschusses mit der gemischten Projektgruppe, die sich aus drei Analytikern, Programmierern sowie drei Benutzern der Fachabteilung zusammensetzt, die Ziele für den nächsten Projektabschnitt. Man einigt sich darauf, die bestehenden Aufgaben und Arbeitsabläufe, die Informationsflüsse und Datenstrukturen sowie die technischen Ausführungsbedingungen in der Fachabteilung zu analysieren. Die Analyse soll in 16 Kalenderwochen durchgeführt und ergebnisorientiert dokumentiert werden. Das methodische Vorgehen bleibt – unter Berücksichtigung firmeninterner Standards – weitgehend der Projektgruppe überlassen.

Dieses Aufgabenpaket und der Zeitplan werden vom Projektleiter dem Lenkungsausschuss vorgelegt und von diesem genehmigt. Der Projektleiter informiert daraufhin die Fachabteilung über das Vorhaben, nimmt Anregungen aus der Fachabteilung auf und meldet diese der Projektgruppe zurück. Die Arbeiten werden nach internen Qualifizierungsmassnahmen von der Projektgruppe aufgenommen.

Der Projektleiter wird im Abstand von 4 bis 5 Wochen über den Stand der Arbeiten informiert, Probleme werden thematisiert, der Projektleiter gibt Anregungen für die weitere Arbeit. Nach Ablauf der 16. Kalenderwoche setzen sich die Projektgruppe und der Projektleiter wiederum zusammen, um den Stand der Analysen, deren Ergebnisse sowie das weitere Vorgehen zu besprechen. Der Lenkungsausschuss wird im Anschluss daran durch den Projektleiter informiert.

Die grenzregulierende Funktion des Projektleiters umfasst weitere Aufgaben wie die Koordination verschiedener Projekt- oder Arbeitsgruppen auf operativer Ebene. Die Schnittstelle zwischen strategischer und operativer Ebene im Projekt ist ebenfalls im Projektleiter personifiziert. In dieser Rolle kommt dem Projektleiter auch eine politische Funktion zu. Er hat das Projekt 'nach aussen' zu vertreten, für das Vorgehen und die Lösungsansätze zu werben bzw. diese – falls notwendig – zu verteidigen sowie zu konsensorientierten Entscheidungen und Lösungswegen im Projekt zu verhelfen. Die Aufgaben des Projektleiters sind in sozialer wie fachlicher Hinsicht anspruchsvoll; die für die erfolgreiche Aufgabenerfüllung notwendigen Qualifikationen sind vielschichtig. Eine angemessene Ausübung dieser Funktion setzt voraus, dass er mit entsprechenden Entscheidungskompetenzen

ausgestattet ist. Die Wahl des Projektleiters erfordert besondere Sorgfalt, das Bereitstellen von Qualifizierungsmöglicheiten sowie ein Verständnis für diese Rolle, das Fehler zulässt bzw. auf Lern- und Entwicklungsfähigkeit aller Beteiligten setzt.

Folgende Qualifikationen und Bedingungen sind für die Arbeit des Projektleiters nach dem skizzierten Verständnis wichtige Voraussetzungen (vgl. Bartsch-Spörl 1991).

Fachkompetenz: Fachbereich- und EDV-Wissen

Der Projektleiter benötigt sowohl ausreichendes Wissen über den Fachbereich, das heisst, dessen Aufgaben und technisch-organisatorische Ausführungsbedingungen als auch fundiertes Wissen über Möglichkeiten und Grenzen der EDV-Technik.

Methodenkompetenz

Der Projektleiter braucht ebenso fundiertes Wissen über Anwendungsmöglichkeiten und Nutzen sowie Grenzen verschiedener Methoden zum Projektmanagement und zur Softwareentwicklung, insbesondere auch über Methoden zur Analyse, Darstellung, Produktion und Evaluation.

Sozialkompetenz

Der Projektleiter benötigt viel Wissen und Erfahrung im Umgang mit Menschen; dies umfasst u.a. Wissen und Erfahrung über Gruppendynamik, Kommunikation und Führungsverhalten. Da Softwareprojekte sehr unterschiedliche Interessengruppen berühren, sind soziale Konflikte oft 'vorprogrammiert'. Es ist die Aufgabe des Projektleiters, mit derartigen Konflikten konstruktiv umzugehen, indem er interessenausgleichende Lösungswege aufzeigt bzw. herbeiführt.

Positives Image und Unterstützung

Der Projektleiter sollte bei den Projektbeteiligten über ein positives Image verfügen, damit er durch die verschiedenen Interessengruppen in seiner Arbeit unterstützt wird und auch seine Funktion als Verbindungsglied zwischen den Interessengruppen ausüben kann.

Identifikation und Empathie

Der Projektleiter muss sich mit den Zielen und Inhalten des Projektes identifizieren, damit er das Projekt wirkungsvoll und glaubwürdig als Promotor vertreten kann.

Unter Empathie ist Einfühlungsvermögen und eine offene Wahrnehmung zu verstehen für das, was im Projekt geschehen kann, real geschieht oder auch nicht geschieht. Mögliche Gefahren sollten dem Projektleiter bekannt sein und entsprechende Warnsignale von ihm frühzeitig, richtig und vollständig erkannt und gedeutet werden (siehe Übersicht 1 im Anhang).

Aus verschiedenen Erfahrungsberichten von Softwareprojekten sind vor allem folgende Gefahren und Warnsignale bekannt:

(1) Man überschätzt sich mit dem Projekt bzw. nimmt sich zuviel vor.

(2) Die Projektziele sind unklar oder schliessen sich gegenseitig aus.

(3) Dem Projekt fehlt ein (Macht-)Promotor.

(4) Die Komplexität der fachtechnischen Aufgabenstellung wird unterschätzt.

(5) Das Projekt ist 'angeordnet' und vom Fachbereich nicht gewollt.

(6) Das Projekt wird kosten- und terminbezogen viel zu knapp budgetiert.

(7) Die organisatorischen Gegebenheiten im Fachbereich werden vor der Systemkonzeption nicht hinterfragt.

(8) Es fehlen angemessene Methoden und Werkzeuge.

(9) Der Zeitplan 'gerät schon sehr früh ins Wanken'.

(10) Die Qualifikationen der Mitglieder im Team sind unzureichend.

(11) Die Ergebnisse werden mangelhaft dokumentiert.

(12) Der Softwareentwurf lässt keine nachträglichen Anforderungsänderungen zu.

(13) In der Projektorganisation fehlen die Benutzervertreter.

(14) Die Qualitätssicherung wird vernachlässigt.

(15) Der Projektgruppe fehlt es an Motivation.

Im folgenden Abschnitt werden vorwiegend Gestaltungshinweise für die Arbeit in der Projektgruppe gegeben.

2.3.4 Die Projektgruppe

In vielen modernen Projektmanagementbüchern wird übereinstimmend Team- bzw. Gruppenarbeit als die adäquate Arbeitsorganisation für die Abwicklung von Projekten dargestellt. Die Ergebnisse einer Meta-Analyse zu Projekterfolgskriterien (Gemünden 1990) zeigen, dass sehr verschiedene Merkmale einer Projektgruppe wie z.B. Fachkompetenz, Entscheidungskompetenzen, Partizipation, Motivation, Kontinuität, Fluktuation im Zusammenhang mit Projekterfolgskriterien stehen. Das Verständnis darüber, wie Teamarbeit zu gestalten ist, damit eine kompetente und motivierte Aufgabenerfüllung erfolgt, ist allerdings in der betrieblichen Praxis sehr unterschiedlich.

In Abschnitt 2.3.1 von Teil B wurde bereits darauf hingewiesen, dass in traditionellen Organisationsmodellen die Arbeit in Projektgruppen häufig funktional und sehr arbeitsteilig gestaltet ist. In grossen Projekten werden z.T. Teams für Analyse und Konzeptentwicklung, andere für die Spezifikation und Realisierung und wiederum andere für die Qualitätssicherung eingerichtet. In kleineren Projekten dagegen wird zwar häufig nur eine Projektgruppe eingesetzt, die die gesamten Aufgaben des Softwarelebenszyklus bearbeitet; solche Gruppen sind jedoch häufig intern funktionsorientiert und arbeitsteilig ausgerichtet und zusammengesetzt (vgl. Strohm & Ulich 1991). Programmierer werden vorwiegend mit programmiertechnischen und testenden Aufgaben beauftragt, dagegen selten mit analytischen und konzeptionellen Aufgaben, die primär zum Aufgabenbereich des Projektleiters und Systemanalytikers gehören.

Programmierer, die unter solch arbeitsteiligen Bedingungen arbeiten, sind mit ihrer Arbeitssituation häufig sehr unzufrieden, da für sie Kooperationsmöglichkeiten weder mit anderen Programmierern noch mit Benutzern speziell in der Analyse- und Konzeptionsphase bestehen. Durch mangelnden Überblick über eine komplexe Aufgabe erleben sie ihre Arbeit als partialisiert und entfremdet (vgl. Hacker 1989). Dies ist gut nachvollziehbar, wenn man bedenkt, dass Softwareentwickler selbst vor allem die analytischen und konzeptionellen Aufgaben – im Unterschied zu den realisierenden Aufgaben – als fachlich und sozial herausfordernd und somit qualifizierend erleben (vgl. Frese et al. 1991)!

Aus starker Funktions- und Arbeitsteilung in und zwischen Projektgruppen resultieren Mangel an Transparenz, Eigeninitiative und Verantwortungsübernahme bei den Gruppenmitgliedern, die Lern- und Entwicklungsmöglichkeiten bei der Arbeit sind eingeschränkt, geringe Motivation und Pro-

duktivität können die Folge davon sein. Dies ist in der betrieblichen Praxis offensichtlich erkannt worden, was sich u.a. daran zeigt, dass verstärkt Analytiker/ Programmierer – die mit der Bearbeitung möglichst vieler Aufgaben des Softwareentwicklungsprozesses beauftragt werden können – ausgebildet bzw. auf dem Arbeitsmarkt gesucht werden.

Effiziente, weitgehend selbständig arbeitende Projektgruppen sind (in Anlehnung an Ulich 1994) durch folgende Merkmale gekennzeichnet:

(1) Relative Unabhängigkeit der Projektgruppe,

(2) innerer Aufgabenzusammenhang in der Projektgruppe sowie

(3) Einheit von Produkt und Projektgruppe.

Dies bedeutet, dass einer Gruppe von Analytiker/Programmierern eine vollständige Projektaufgabe – welche idealtypischerweise möglichst viele interdependente Teilaufgaben von A bis Z umfasst – übertragen wird. In gemischten Gruppen wird eine solche Gruppe z.B. durch Benutzervertreter ergänzt. Die Übertragung einer vollständigen Projektaufgabe ist eine wesentliche Voraussetzung für das Verständnis einer gemeinsamen Aufgabe in der Gruppe. Dieses Verständnis ermöglicht ein höheres Mass an Selbstregulation und gegenseitiger Unterstützung (vgl. Ulich 1994). Selbstregulation beinhaltet dabei, dass die Gruppe neben inhaltlichen Aufgaben auch organisatorische Aufgaben wahrnimmt. Die Organisation der Arbeit erfolgt weitgehend selbständig durch die Projektgruppe.

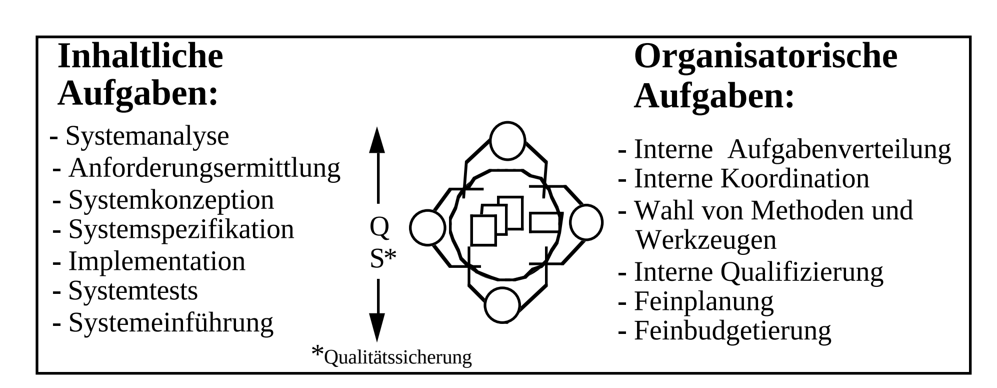


Abbildung 9: Inhaltliche und organisatorische Aufgaben in einer selbstregulierenden Projektgruppe.

In der Abbildung 9 sind inhaltliche und mögliche organisatorische Aufgaben in einer selbstregulierenden Projektgruppe aufgeführt. Aus dieser Abbildung ist ersichtlich, dass in einer selbstregulierenden Projektgruppe neben der ganzheitlichen Projektaufgabe auch organisatorische Aufgaben wie die interne Aufgabenverteilung und Koordination, Feinplanung usw. wahrgenommen und ausgeführt werden. Dies beinhaltet, dass z.B. die Form der internen Koordination durch die Gruppe selbst festgelegt wird.

Im Sinne der arbeitswissenschaftlichen Forderung für die industrielle Produktion, Qualität nicht zu kontrollieren, sondern zu produzieren, ist die *begleitende Qualitätssicherung* über alle Projektschritte des Softwarelebenszyklus eine weitere Aufgabe der Projektgruppe. Dies beinhaltet, dass die Gruppe z.B. auch Reviews zu den Anforderungen und zum Entwurf durchführt. Eine wichtige Voraussetzung für die dargestellte Arbeitsform ist die soziale Integration. Deshalb sollte die Projektgruppe – für die Zeit der Projektarbeit – in einem gemeinsamen Büro oder zumindest räumlich nahe zusammen arbeiten, damit Kommunikation und Kooperation erleichtert werden (siehe auch DeMarco und Lister 1991).

Auf die Rolle des Projektleiters in einer selbstregulierenden Projektgruppe wurde in Kapitel 2.3.3 des Teils B bereits eingegangen. In grösseren Projekten, die den Einsatz mehrerer solcher Gruppen erfordern, ist eine wesentliche Funktion des Projektleiters die Koordination der verschiedenen Gruppen, damit konsistente Teilergebnisse entstehen, die zu einem Ganzen zusammengeführt werden können.

Die Aufteilung von grossen Projekten in kleine, überschaubare und ganzheitliche Teilprojekte im Sinne einer produktorientierten Arbeitsteilung – welche auf der Basis einer modularen Systemkonzeption erfolgen kann – bekommt in diesem Zusammenhang eine zentrale Bedeutung.

In einem sehr grossen Projekt, welches die Einrichtung von z.B. sechs selbstregulierenden Projektgruppen bedingt, ist die Schaffung einer Projektstruktur mit *überlappenden Gruppen* eine Notwendigkeit, damit verteiltes Wissen und verschiedene Arbeitsergebnisse integriert werden können. Die Entwicklung eines Gesamtkonzeptes des Systems oder eines übergreifenden Konzeptes zur Qualitätssicherung wäre zudem ein Beispiel für eine Projektaufgabe, welche in übergeordneten Gruppen ausgeführt werden kann bzw. muss.

Aus Abbildung 10 ist ersichtlich, dass übergeordnete Gruppen aus Vertretern der verschiedenen Projektgruppen gebildet werden. Die Vertreter in

diesen übergeordneten Gruppen werden entweder durch die Gruppe fest bestimmt oder wechseln nach einem Rotationsprinzip. Die Aufgaben dieser übergeordneten Gruppen können Koordination, Wissens- und Ergebnistransfer oder spezifische Aufträge umfassen.

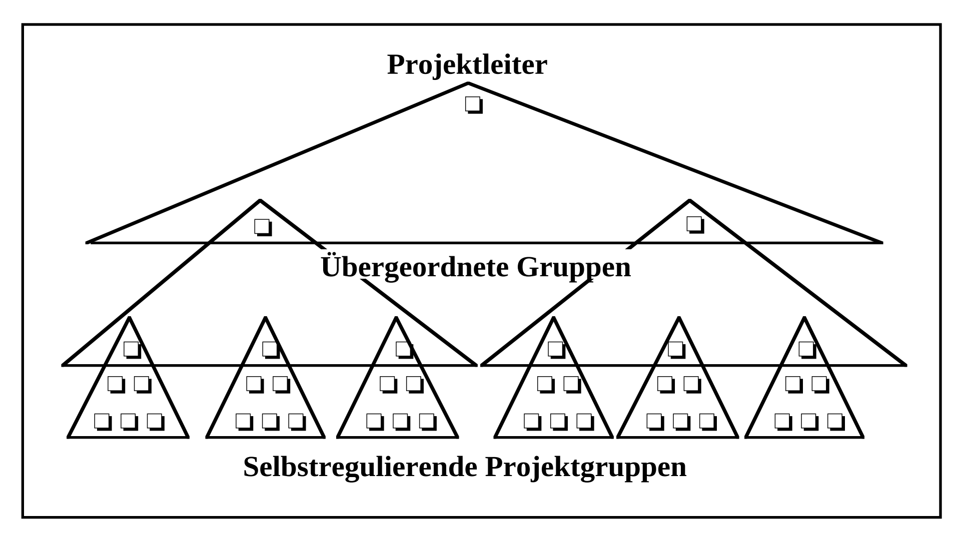


Abbildung 10: Ein Modell überlappender Projektgruppen (in Anlehnung an Likert 1972).

Vertreter mehrerer übergeordneter Gruppen bilden – falls notwendig – mit dem Projektleiter eine weitere Instanz, in der projektübergreifend Absprachen und Koordination stattfinden können. Eine Projektorganisation mit selbstregulierenden Gruppen kann folgende Vorteile zeigen:

(1) Sie fördert die Identifikation der Gruppe mit der Projektaufgabe.

(2) Die Arbeit in derartigen Gruppen ist qualifizierend in fachlicher wie in sozialer Hinsicht.

(3) Die Flexibilität, Kreativität und Motivation in der Projektgruppe wird dadurch gefördert.

(4) Diese Form der Projektorganisation wirkt der Bürokratisierung von Projekten entgegen.

Kontinuität in der personellen Zusammensetzung von Projekten, bei der Qualifikationen und Erfahrungen erhalten bleiben und nicht z.B. infolge von Fluktuationen verlorengehen, wird durch selbstregulierende Projektgruppen ebenfalls wahrscheinlicher bzw. besser erreicht. Damit sich diese Vorteile einstellen, sind jedoch einige Bedingungen zu erfüllen:

(1) Die Projektaufgabe sollte für die Gruppenmitglieder überschaubar sein.

(2) Die einzelnen Teilaufgaben sollten einen inneren Zusammenhang aufweisen.

(3) Die Projektgruppe sollte über vereinbarte Output-Ziele verfügen.

(4) Die Projektgruppe sollte über den Grad der Zielerreichung laufend Feedback erhalten.

Erfahrungsgemäss zeigt sich, dass Arbeits- bzw. Projektgruppen eine Grösse von acht bis zehn Personen nicht übersteigen sollten, damit neben der geforderten aufgabenbezogenen Transparenz auch die Transparenz des 'sozialen Systems' erhalten bleibt. Ergebnisfeedback schon während der Entwicklung ist wiederum am besten durch den Einsatz von Prototyping und eine enge Zusammenarbeit mit den Benutzern zu erreichen.

Eine weitere wichtige Voraussetzung für die Arbeit in selbstregulierenden Projektgruppen ist ein Entlohnungskonzept, das weniger auf einer individuellen Leistungsbemessung basiert als vielmehr die gesamte Gruppenleistung sowie die Qualifikation der einzelnen Gruppenmitglieder – das heisst ihr Können ('pay for knowledge') und damit ihre Flexibilität und Einsetzbarkeit – honoriert. Gruppenleistungsanteile, die z.B. für das Erreichen vereinbarter Qualitäts- oder Budgetziele vorgesehen sind, sollten gleichmässig auf alle Gruppenmitglieder verteilt werden. Dies ist eine weitere Möglichkeit, um die kollegiale Unterstützung in der Projektgruppe zu fördern.

2.4 FAZIT

Im Rahmen dieses Kapitels wurden u.a. die zentralen Elemente des Projektmanagements dargestellt sowie eine soziotechnische Betrachtung der Softwareentwicklung angestellt. Dabei kommt den Aufgaben der Zieldefinition und Projektklassifikation eine zentrale Bedeutung zu. Vor dem Hintergrund der Merkmale und Probleme traditioneller Formen der Aufbauorganisation bei der Softwareentwicklung wurden alternative Modelle vorgestellt. Das Hauptaugenmerk wurde dabei auf projektorganisatorische Mög-

lichkeiten der Fachbereichsvertretung bzw. des Benutzereinbezugs sowie eine Beschreibung der Aufgaben des Projektleiters und der Projektgruppe gelegt. Im Vergleich zu traditionellen Projektmanagementmodellen basieren die beschriebenen Ansätze auf dezentralen Strukturen, Benutzerorientierung und Selbstregulation. Vielfältige Erfahrungsberichte bestätigen den Erfolg dieser Formen der Arbeits- und Organisationsgestaltung (siehe insbesondere DeMarco und Lister 1991).

Dennoch stossen solche Ansätze immer wieder auf Widerstand. Ein Grund dafür liegt sicherlich darin, dass diese Konzepte etablierte Machtstrukturen bedrohen. Ulich (1991) kommt in diesem Zusammenhang zu folgendem Fazit: "Das Problem, um das es sich hier handelt, ist offenbar mindestens so alt wie unsere Zeitrechnung. Es wurde bereits in einer Schilderung benannt, die Petronius, dem römischen Satiriker und Senator zur Zeit Neros, zugeschrieben wird: 'Wir trainierten hart ... aber es schien, dass wir immer dann reorganisiert wurden, wenn wir gerade dabei waren, ein Team zu werden'."

Die Wahl zwischen traditionellen oder alternativen Organisationsmodellen zur Softwareentwicklung ist daher auch unmittelbar mit der folgenden Frage verknüpft: Entweder dient das Projekt der Erhaltung und Zementierung bestehender Arbeits-, Organisations- und Machtstrukturen – mit den entsprechenden Problemen – oder es kann dazu eingesetzt werden, die Entwicklung effizienter, aufgaben- und benutzungsorientierter Software zur Unterstützung innovativer Arbeitsabläufe umzusetzen.

Der Übergang von klassischen, funktional arbeitsteiligen hin zu zeitgemässen, funktional integrierten Arbeits- und Organisationsstrukturen kann aus arbeitspsychologischer Perspektive allerdings nur durch eine Form der Projektarbeit erreicht werden, welche u.a. die Kreativität und Innovativität der Projektmitarbeiter fördert. Die beschriebenen Ansätze können dazu einen Beitrag leisten.

3 Prozessgestaltung

In diesem Kapitel wird eine Bestandsaufnahme der bisher bekannten Problembereiche des traditionellen Phasenmodells (das 'Wasserfall'-Modell) vorgenommen, deren Überwindung zu einem iterativ-zyklischen Ablaufmodell führt. Dieses iterativ-zyklische Ablaufmodell (Quadrantenmodell) für partizipative Softwareentwicklungen macht aufgrund seiner zyklischen Struktur neue Vorgehensweisen notwendig, welche mit modernen Konzepten der Qualitätssicherung bei der Softwareentwicklung übereinstimmen.

3.1 Warum das Phasenmodell nicht genügt

Das traditionelle, lineare Phasenmodell der Softwareentwicklung (Frühauf, Ludewig und Sandmayr 1991a) unterscheidet die folgenden Phasen:

(1) Anforderungsanalyse,

(2) Aufgaben- und Systemanalyse,

(3) Spezifikation der Anforderungen,

(4) Systementwurf ggf. Modulspezifikation,

(5) Codierung ggf. Modultest,

(6) Integration und Systemtest,

(7) Installation und Abnahmetest,

(8) Betrieb und Wartung.

In diesem linearen Ablaufmodell ist es bei der Entdeckung von Fehlern, Problemen usw. nur möglich, zu der jeweils unmittelbar vorhergehenden Phase zurückzukehren. Oftmals werden jedoch auch Fehler aus früheren Phasen entdeckt. Die bekanntesten Schwächen des traditionellen Phasenmodells sind in Kasten 18 beschrieben.

Die Attraktivität des Phasenmodells kommt daher, dass sich die einzelnen Phasen scheinbar logisch direkt auf der linearen Zeitachse abbilden lassen. Somit wird das Phasenmodell aber als Projektmanagementmethode zur Steuerung des Entwicklungsprozesses missverstanden. Dabei dienen die Phasenergebnisse als 'Meilensteine', ein beliebtes – wenn auch häufig unzureichendes – Controlling-Instrument. Für eine angemessene Ablauforganisation wird jedoch ein der inhaltlichen Struktur des Softwareentwick-

lungsprozesses angepasstes, von der Zeitdimension zunächst unabhängiges Modell benötigt. Erst in einem zweiten Schritt wird die zyklische Struktur hinsichtlich der notwendigen, unterschiedlichen Aktivitäten der unternehmensspezifischen Projektkultur auf der linearen Zeitachse abgebildet (siehe auch AMI 1992). Um diese Abstraktion zu begründen, werden Konzepte aus der Systemtheorie benötigt.

Kasten 18: Die bekanntesten Schwächen des traditionellen Phasenmodells (aus Hesse, Merbeth und Frölich 1992, S. 75).

> "Häufig keine systematische Benennung und Trennung von Tätigkeiten, Ergebnissen und Phasen; keine über zwei benachbarte Phasen hinausgehenden Entwicklungszyklen; keine Querbezüge zwischen frühen und späten Entwicklungsphasen; lange Entwicklungsdauer zwischen Vorgaben (Spezifikation) und Realisierung des Systems bzw. seiner Bausteine; keine oder ungenügende Beachtung moderner Werkzeuge mit weitreichenden Generierungsmöglichkeiten, die den gesamten Projektablauf verändern; keine oder unzureichende Behandlung des Wartungsproblems; keine Vorkehrungen für die Kooperation zwischen Anwendern und Entwicklern, die Einbeziehung der Benutzer in den Entwicklungsprozess und evolutionäre Systementwicklung."

In der Systemtheorie wird zwischen den beiden Lenkungsprinzipien der 'Steuerung' und der 'Regelung' unterschieden. Das traditionelle Phasenmodell beruht primär auf dem Prinzip der 'Steuerung': Arbeitsaufträge werden von einer Phase an die nächste vergeben. Dabei wird davon ausgegangen, dass alle wesentlichen Voraussetzungen für die Auftragsbearbeitung in der folgenden Phase bereitstehen und es keine ungeplanten bzw. unvorhergesehenen Einflüsse gibt (z.B. mangelhafte Hardware, fehlende Ressourcen, unzureichende Qualifikation, Aktivitäten höherer Priorität usw.). Die Projektleitung hat ein komplettes Wissen über alle geplanten Folgeaktivitäten; es gibt drei Voraussetzungen für dieses 'komplette Wissen':

(1) Die genaue Kenntnis der Reaktionen aller technischen und sozialen Komponenten,

(2) die genaue Kenntnis aller, die geplanten Aktivitäten beeinträchtigenden Grössen ('Störgrössen') sowie

(3) ein hinreichendes Wissen über die notwendigen Interventionen, um allen Störgrössen adäquat und *im vorhinein* begegnen zu können.

Diese drei Voraussetzungen sind – selbst bei sehr erfahrenen Projektleitern – praktisch nie vollständig erfüllt (Schiemenz 1979, S. 1024).

Die Verwendung des sehr leistungsfähigen Lenkungsprinzips der 'Regelung' dagegen setzt im Grunde nur die Kenntnis der Aufträge voraus, die sich auf das Ergebnis einer Projektaktivität der Tendenz nach in die gewünschte Richtung auswirken. Übertragen wir das allgemeine Schema für Regelung auf den Kontext der Softwareentwicklung, so gelten als Optimalitätskriterien alle relevanten technischen und sozialen Faktoren. Der Soll-Ist-Wertabgleich bei der 'Regelung' überprüft, inwieweit die von aussen vorgegebenen Soll-Werte unter Einhaltung der Randbedingungen erfüllt sind. Dies kann durch die Grenzregulationsaufgaben der Projektleitung bewerkstelligt werden.

Natürlich kommt auch das Lenkungsprinzip der 'Steuerung' in aktuellen Softwareentwicklungen zur Anwendung. Hierbei ist an direktive Entscheidungen und Vorgaben durch den Auftraggeber, die Projektleitung oder andere Führungsgremien usw. zu denken. Oft arbeiten Steuerungssysteme wirtschaftlicher als entsprechende Regelungssysteme – *aber nur*, wenn die oben genannten Voraussetzungen zutreffen! Dies ist auch der Grund dafür, warum grundsätzlich in erster Linie versucht wird, ein gesteuertes System herzustellen, sprich: dem Phasenmodell so nahe wie möglich zu kommen. Nehmen wir jedoch die empirisch ausreichend belegte Tatsache *ernst*, dass niemand im vorhinein (z.B. vor Projektbeginn) ein hinreichendes Wissen über den zu 'steuernden' Prozess (z.B. komplexe, innovative Projekte) haben kann, so gilt es zu entscheiden, an welchen Stellen im Softwareentwicklungsprozess Optimierungszyklen *unabdingbar* sind.

3.2 DER OPTIMIERUNGSZYKLUS

Die zyklische Struktur des Regelkreises lässt sich nun mit einigen Anpassungen zu einem *Optimierungszyklus* im Rahmen der Softwareentwicklung ausbauen. Die Grundidee besteht darin, dass alle *Aktivitäten* durch eine entsprechende *Testkomponente* ergänzt werden (AMI 1992). Dies klingt zunächst aufwendiger, als es tatsächlich ist. Viele 'Tests' werden bereits

heute schon angewendet, ohne dass man sich dessen oftmals bewusst ist. Wir werden weiter hinten sehen, welche 'Tests' dies im Einzelnen sind.

Als 'Aktivität' können sehr unterschiedliche Verfahren, Methoden und Techniken in Frage kommen; dies richtet sich nach der Art des Ergebnisses. Während die 'Randbedingungen', welche von aussen auf einen Optimierungszyklus einwirken, bewusst und zielorientiert vorgegeben werden (z.B. Budget, Projektdauer usw.), sind die 'Störgrössen' unbeabsichtigt und unvorhersehbar. Wir bezeichnen den 'Test-Aktivitäts-Zyklus' als *Optimierungszyklus* (siehe Abbildung 11). Eine wesentliche Beschreibungsgrösse des Optimierungszyklus ist die benötigte Zeit für einen Durchlauf. Je grösser die Durchlauf-, das heisst, Zykluszeit ist, desto aufwendiger ist der jeweilige Optimierungszyklus. Ziel benutzerorientierter Softwareentwicklung ist es nun, möglichst effiziente Optimierungszyklen in den Entwicklungsprozess einzubauen.

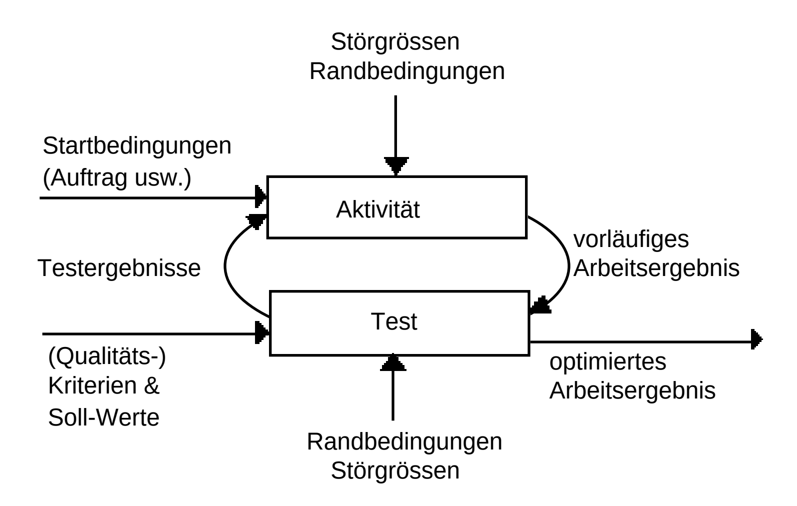


Abbildung 11: Die einzelnen Komponenten des Optimierungszyklus in Anlehnung an Deming (1989).

In den folgenden Abschnitten dieses Kapitels werden die einzelnen Abschnitte und Phasen von benutzerorientierter Softwareentwicklung vorgestellt und die in den einzelnen Phasen möglichen Aktivitäten zur Beteiligung von Benutzern diskutiert. Eine benutzerorientierte Softwareentwicklung

lässt sich am besten im Rahmen eines zyklischen Versionenkonzeptes realisieren. Wir teilen den gesamten Versionenzyklus in vier Quadranten auf:

Quadrant-I umfasst die Analyse und Bewertung des Arbeitssystems, in dem die Software eingesetzt werden soll; aus der Bewertung werden dann die Anforderungen abgeleitet. Durch die mehrfache Optimierung im *Quadrant-I* entsteht aus dem 'Grobkonzept' somit das 'Feinkonzept'.

Quadrant-II umfasst die Spezifikation (z.B. das Datenmodell) und den Entwurf, welcher dann im

Quadrant-III als Programm realisiert wird. Nach einer ersten Testphase der einzelnen Programmteile erfolgt die Integration; die auslieferbare Version wird dann im

Quadrant-IV im Arbeitssystem eingesetzt und benutzt.

3.3 QUADRANT-I: DIE ANALYSE- UND BEWERTUNGSPHASE

Als erstes muss bestimmt werden, 'ob' und 'wo' ein sinnvoller Einsatz von Technologie möglich bzw. erforderlich ist. "Die Vorstellung, man könne mit Hilfe der Technik die Defizite einer Organisation beheben, ohne die Strukturen der Gesamtorganisation in Frage zu stellen, ist zwar noch immer weitverbreitet, aber zumeist ein Trugschluss" (Klotz 1991, S. 108). Wichtig ist, das Arbeitssystem als eine *lebendige Organisation*, als einen sich selbst erhaltenden Organismus zu begreifen, der sich entwickeln und verändern muss, um die Organisationsziele zu erreichen. Unter dieser Perspektive geht es bei der Festlegung der Organisationsschnittstelle primär darum, die Lebensfähigkeit der Organisation durch Einsatz von Technologie zu fördern bzw. zu erhalten. Hierzu wird ein fundiertes Wissen für arbeits- und organisationsgestalterische Massnahmen benötigt (siehe Übersicht 3 im Anhang).

Alle Bemühungen – seitens der Softwaretechnik – zur *Prozessmodellierung* können nicht an der Tatsache vorbeigehen, dass die wesentlichen Aktivitäten in dieser Phase *Prozessmoderationen* sind. Gerade das Agieren im organisatorischen Umfeld ist von entscheidender Bedeutung. Erst eine geeignete Prozessmoderation erlaubt es dem Unternehmen, selbst herauszufinden, welche Veränderungen wünschenswert, notwendig und möglich sind.

Da die meisten Softwareentwicklungsprojekte primär von Softwaretechnikern durchgeführt werden, diesen aber oft eine ausreichende Ausbildung in

Prozessmoderation fehlt, ist die Analysephase oft die am meisten vernachlässigte Phase. In der angewandten Psychologie wurden und werden primär kommunikative Methoden und Techniken zur Prozessmoderation entwickelt, welche dann in der Arbeits- und Organisationspsychologie übernommen, angepasst und angewendet werden. Der Nutzen dieser Methoden wird von Softwareentwicklern nach eigenen Angaben häufig unterschätzt.

Die Fehlerbehebungskosten in späteren Phasen, die durch eine suboptimale Analyse entstehen, sind im Vergleich zu einem grösseren Analyseaufwand unverhältnismässig hoch: Faktor 100 bis 1000! (siehe dazu Boehm 1981). Es ist an der Zeit, für eine optimale Softwareentwicklung speziell ausgebildete Personen in Prozessmoderationsmethoden für die Analyse- und Bewertungsphase einzusetzen.

3.4 QUADRANT-II: DIE SPEZIFIKATIONSPHASE

Im Anschluss an die Analyse und Bewertung des Arbeitssystems gilt es die gewonnenen Ergebnisse in eine implementierungsnahe Form zu transformieren. Hierzu sind Spezifikationsmethoden mit hohem kommunikativem Wert einzusetzen (z.B. 'RFA'-Netze von Oberquelle 1987 und andere Verfahren: Martin und McClure 1985, Martin 1988, Wood und Silver 1989, August 1991, Raasch 1991).

Die Spezifikation der Werkzeugschnittstelle:

Bei der Spezifikation der Werkzeugschnittstelle wird die intendierte Festlegung zur Mensch-Computer-Funktionsteilung vorgenommen; insbesondere geht es darum, die für die Aufgabenbearbeitung entscheidungsrelevanten Informationen und die dazu notwendigen Arbeitsobjekte mit den entsprechenden Funktionen herauszufinden (Diaper 1989; Dunckel et al. 1993; siehe auch Übersicht 3 im Anhang). Die Aufgaben, welche beim Menschen verbleiben, sollten sich durch die in Teil A, Kapitel 1.2 aufgeführten Merkmale der Aufgabengestaltung auszeichnen.

Die Spezifikation der Ein-/Ausgabeschnittstelle:

Nachdem (einigermassen) Klarheit unter allen Beteiligten darüber besteht, welche Funktionen automatisiert werden, empfiehlt es sich, zunächst mit den Endbenutzern das Bildschirmlayout, z.B. mittels Handskizzenentwurf, zu entwerfen und auszutesten. Bei sehr umfangreichen Mengen unterschiedlicher Masken kann eine Bilddatenbank die mittels Grafikeditor erstellten Masken verwalten. Der Einsatz von Prototypingwerkzeugen für diesen Zweck ist oft deshalb unangemessen, weil die 'tool'-spezifischen

Darstellungsmöglichkeiten oft zu begrenzt sind. Die Auswirkungen der umgesetzten Gestaltungsmassnahmen lassen sich durch Expertenevaluation, Diskussionen mit den Endbenutzern, z.B. im Rahmen von Workshops bzw. mittels Check-Listen (siehe Übersicht 2 im Anhang), austesten, überarbeiten und optimieren.

Die Spezifikation der Dialogschnittstelle:

Zur Spezifikation der Dialogschnittstelle ist es unabdingbar, Prototypen zur Veranschaulichung der dynamischen, interaktiven Aspekte des zu entwikkelnden Werkzeuges einzusetzen. Prototypen sollten nur ganz gezielt und zweckgebunden zur Abklärung spezieller Spezifikationsaspekte eingesetzt werden, weil ansonsten die Gefahr besteht, zuviel Aufwand in den Ausbau und den Erhalt von 'Anschauungsprodukten' zu investieren. Eine sehr effiziente und zudem preiswerte Variante ist die Verwendung von Simulationsstudien, z.B. mittels handgemalter Folien usw., welche dem Benutzer in Abhängigkeit von seinen Aktionen von einem Testleiter, welcher das geplante Systemverhalten ausreichend kennt, vorgelegt werden. Die Auswirkungen der umgesetzten Gestaltungsmassnahmen lassen sich z.B. mittels spezifischer Tests (siehe Teil B, Kapitel 4) oder mit Checklisten (siehe Übersicht 2 im Anhang) austesten, überarbeiten und optimieren.

3.5 QUADRANT-III: DIE REALISIERUNGSPHASE

Nachdem in der Analyse- und Spezifikationsphase der notwendige Optimierungsaufwand geleistet wurde, kann die Realisierungsphase beginnen. Die Realisierungsphase gliedert sich in die folgenden drei Schritte:

(1) Entwurf der Programm-Architektur; (2) Entwurf der einzelnen Programm-Module (-Objektklassen usw.); (3) Codierung und 'Debugging'.

Es ist wichtig, den Entwurf von der Spezifikation zu unterscheiden. Während in der Spezifikation möglichst präzise alle relevanten Eigenschaften des technischen Teilsystems festgelegt werden, muss in der Realisierungsphase dafür Sorge getragen werden, dass das zu entwickelnde technische Teilsystem möglichst alle diese Eigenschaften aufweist. Hier kommen primär rein softwaretechnische Kenntnisse zum Tragen: einfache und klare Systemstruktur, Entkopplung von Bausteinen, Minimalität und Abstraktheit von Schnittstellen, Vermeidung globaler Daten, Bildung abstrakter Systemsichten usw. (Sommerville 1989; Ott 1991; Frühauf, Ludewig und Sandmayr 1991a).

Vor der eigentlichen Codierungsphase ist zu prüfen, inwieweit sich schon vorhandene Software wiederverwenden lässt. Sollte dies nur beschränkt oder gar nicht möglich sein, ist zu testen, welche Programmier- bzw. Entwicklungsumgebung ('CASE workbench') die grösstmögliche Effizienz verspricht. Zu einer Entwicklungsumgebung gehört zunächst im Zentrum ein Sammelbehälter ('information repository'). In diesen Sammelbehälter werden Dokumente und Produkte mittels verschiedener Werkzeuge abgelegt bzw. entnommen: das 'data dictionary', Werkzeuge zum Suchen im Sammelbehälter, Werkzeuge zur Erzeugung strukturierter Diagramme, Werkzeuge zur Entwurfsanalyse und -testung, entwurfsgesteuerte Programmgeneratoren, Maskengeneratoren, Reportgeneratoren, Import- und Exportmöglichkeiten. Durch den Einsatz derartiger Entwicklungsumgebungen lassen sich Produktivitätssteigerungen bis zu 40% erreichen (Chikofsky und Rubenstein 1988).

Die einzelnen Programmteile bzw. die integrierte Version lassen sich mittels Alpha- und Beta-Tests hinsichtlich Korrektheit, Performanz usw. austesten (siehe Frühauf, Ludewig und Sandmayr 1991b).

3.6 QUADRANT-IV: DIE IMPLEMENTATIONS- UND BENUTZUNGSPHASE

Nach Erstellung einer lauffähigen Version kann diese zu Benutzbarkeitsstudien (benutzungsorientierte Benutzungstests, Pilot-Benutzer) im konkreten Arbeits- und Aufgabenkontext eingesetzt werden. Erst zu diesem Zeitpunkt lassen sich alle Probleme mit dem aktuellen organisatorischen und technischen Umfeld abklären. Diese Feldstudien tragen im Unterschied zu Laborstudien dem Aspekt der 'ökologischen Validität' Rechnung. Durch die konkrete Benutzung im realen Aufgabenkontext kann der erreichte Anpassungsgrad an den geplanten organisatorischen Soll-Zustand mittels Interview, Fragebogen, Benutzungstest usw. überprüft und getestet werden. Daten über die Benutzung lassen sich auf Video aufnehmen, durch den Testleiter protokollarisch festhalten oder automatisch in Logfiles abspeichern. Obwohl Videoaufnahmen die informationsreichsten Datenaufzeichnungen sind, hat sich eine Kombination aus Logfile und direkter Protokollierung als ein guter "Kompromiss zwischen Leistungsfähigkeit und Ökonomie" herausgestellt (Müller-Holz et al. 1991).

3.7 DAS ITERATIV-ZYKLISCHE ABLAUFMODELL

Die beim Start und beim Durchlauf einzelner Optimierungszyklen zu beachtenden Aspekte werden im folgenden diskutiert (siehe auch Hesse, Merbeth und Frölich 1992). Als eine wesentliche Rahmenbedingung von Softwareentwicklung hat sich z.B. der Typ der zu entwickelnden Software herausgestellt (Typ A, B, C, D; siehe Teil B Kapitel 2.2).

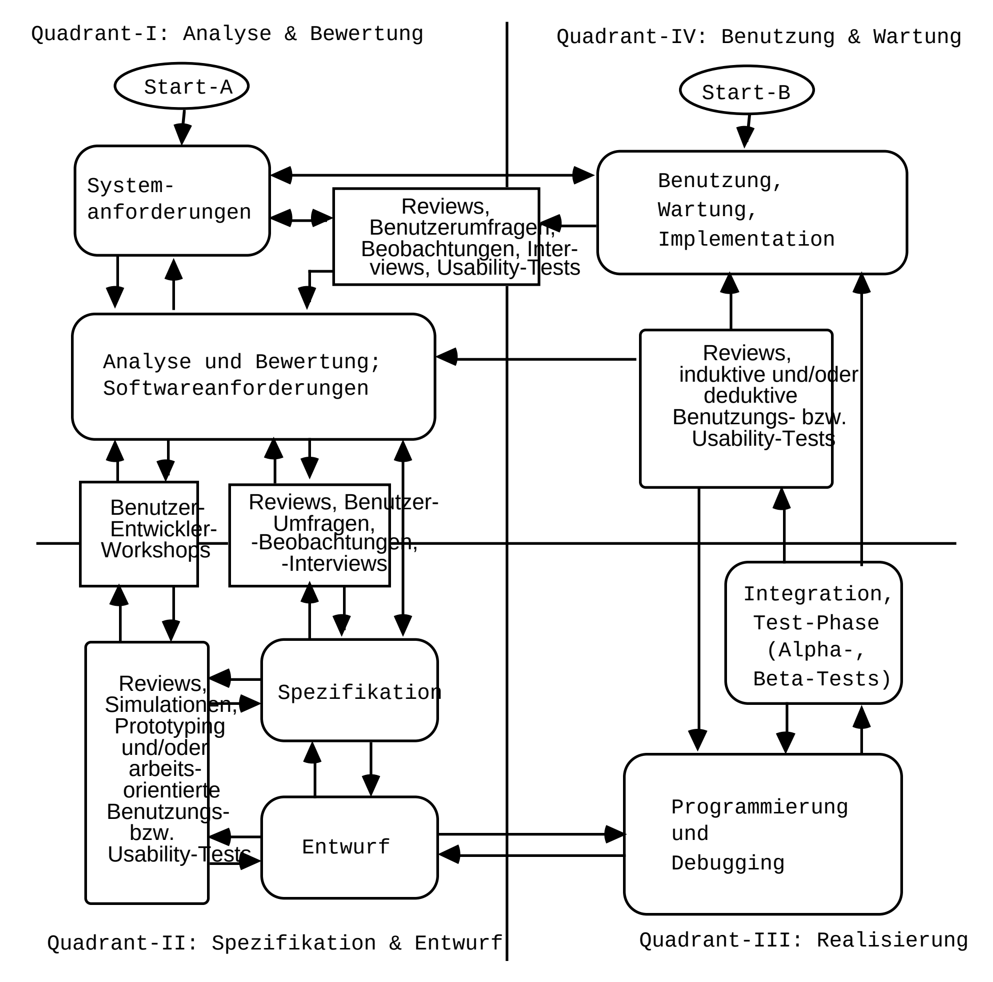


Abbildung 12: Ablaufmodell eines partizipativen Softwareentwicklungskonzeptes mit einer Übersicht über die verschiedenen Optimierungszyklen innerhalb und zwischen den einzelnen Quadranten.

Der Einstieg in den globalen Optimierungszyklus (siehe Abbildung 12) bei einer Neuentwicklung erfolgt über den Start-A, während bei einer Weiterentwicklung der Einstieg bei Start-B erfolgt. Je nach Projekttyp kommen kontextspezifisch unterschiedliche lokale Optimierungszyklen zur Optimierung spezifischer Arbeitsergebnisse zum Einsatz. Die Entscheidung über die jeweils aktuelle Vorgehensweise gehört zur Aufgabe des Projektmanagements und schlägt sich in der gewählten Aufbauorganisation nieder (vergleiche Teil B, Kapitel 2).

Der globale Optimierungszyklus mit seinen eingebetteten lokalen Zyklen lässt sich in vier Bereiche unterteilen (Quadrant-I bis -IV, siehe Abbildung 12). Quadrant-I umfasst die Analyse und Bewertung des Arbeitssystems, in dem die Software eingesetzt werden soll. Hierbei werden überwiegend kommunikative, informale bzw. semiformale Methoden der Arbeits- und Organisationspsychologie verwendet. Im Quadrant-II wird die (Detail-) Spezifikation z.B. unter Einsatz von Prototypen optimiert. In diesem zweiten Quadranten werden verstärkt formale Methoden eingesetzt.

Im Quadrant-III wird die spezifizierte und entworfene Software programmiert und an Testdaten überprüft. Der Quadrant-IV umfasst die Testung und Optimierung des entwickelten Systems in der realen Einsatzumgebung. Die Rückkopplung von der Benutzungsphase (Quadrant-IV) zur Analyse- und Bewertungsphase (Quadrant-I) ermöglicht ein Vorgehen nach dem Versionenkonzept.

Das traditionelle Phasenmodell ist als Spezialfall in dem hier vorgestellten Ablaufmodell enthalten: Der Softwareentwicklungsprozess verläuft weitgehend ohne Rückkopplungen von Start-A über die System- und Softwareanforderungen, die Spezifikationsphase in den Entwurf und die Programmierung; danach wird das Softwaresystem ausgetestet und im Arbeitssystem implementiert. Ohne Rückkopplung zum Quadrant-I wird das Projekt beendet.

Je nach Projekttyp und -auftrag wird der Optimierungsaufwand in den einzelnen Quadranten unterschiedlich ausfallen. Alle bisher durchgeführten Analysen von Softwareentwicklungsprojekten zeigen jedoch, dass mit zunehmendem Optimierungsaufwand in Quadrant-I der Wartungsaufwand in Quadrant-IV abnimmt und insgesamt Kosten gespart werden. Je weniger Aufwand jedoch in Quadrant-I investiert wird, desto mehr Anpassungs- bzw. Wartungsaufwand fällt in Quadrant-IV an. Dieser seit über zehn Jahren bekannte Zusammenhang ist nach wie vor gültig. Tatsächlich ist es eine irrige Vorstellung, man könne die ungelösten Probleme des ersten

Quadranten ohne weiteres im vierten Quadranten aufarbeiten (Boehm 1981, S. 40).

4 Vorgehensweisen und Methoden in der Praxis

4.1 ÜBERSICHT

Wie eingangs im letzten Kapitel vor dem Hintergrund systemtheoretischer Betrachtungen aufgezeigt wurde, setzt sich jeder einzelne Optimierungszyklus aus einer Test- und einer Aktivitätskomponente mit entsprechender Verkopplung zusammen. Beide Komponenten können recht unterschiedlicher Art sein. In der Tabelle 5 geben wir eine Übersicht über Einsatzschwerpunkte, Arten der Aktivität und Tests, das jeweilige Arbeitsergebnis sowie den geschätzten Bereich der Zyklusdauer. Je kürzer die Zyklusdauer ausfällt, desto schneller und damit öfter kann der Optimierungszyklus durchlaufen werden. Je öfter ein Optimierungszyklus durchlaufen wird, desto eher kann ein optimales Ergebnis erreicht werden.

Für die Festlegung des Zeitrahmens der einzelnen Aktivitäten kann man nun anhand der Zyklusdauer der einzelnen Optimierungszyklen ausrechnen, wieviel Optimierung zu welchem Preis machbar und ggf. erstrebenswert ist. Sind bei der Projektetablierung die benötigten Optimierungszyklen nach inhaltlichen Gesichtspunkten festgelegt, so kann man ausrechnen, wieviel Zeit in etwa für die einzelnen Optimierungsaktivitäten benötigt wird: nämlich mindestens die Dauer des ersten Zyklus. Wieviele Zyklusdurchläufe notwendig werden, hängt von den Ergebnissen der jeweiligen Testphase ab; ab zwei Zyklusdurchläufen kann man schon abschätzen, wie schnell die 'gemessene' Produktqualität sich nach diesen zwei Optimierungszyklen dem wahrscheinlich noch nicht erreichten Soll-Wert annähert.

Ein wesentliches Problem der adäquaten Verzahnung der verschiedenen Optimierungszyklen besteht in ihrer *Synchronisation*. Sind an verschiedenen Stellen im Quadrantenmodell (siehe Abbildung 12) gleichzeitig mehrere Optimierungszyklen aktiv, so müssen diese adäquat synchronisiert werden. Dieser Aspekt ist deshalb besonders wichtig, weil nur so das Ausmass an Inkonsistenzen innerhalb des gesamten Entwicklungsprozesses minimiert werden kann. Wenn z.B. parallel zur Realisierungsphase weitere Rücksprachen und Anforderungsanalysen mit dem Anwender stattfinden, passiert es leicht, dass die Entwickler gemäss den stets veralteten Spezifikationen oftmals für den 'Papierkorb' programmieren. Die Ursache hierfür ist

bei fehlender oder mangelnder Synchronisation dieser beiden Optimierungszyklen in ihrer unterschiedlichen Zyklusdauer zu sehen.

Methode	**Aktivität**	**'Test'**	**Ergebnis**	**Zyklus-Dauer**
Diskussion (siehe Teil B, Kapitel 4.5)	verbale Kommunikation Metaplan, Flipcharts usw.	rein verbale Interpretation visuelle + verbale Interpretation	globale Design-Entscheidungen spezifische Design-Entscheidungen	Sekunden – Minuten Minuten – Stunden
Simulation (siehe Teil B, Kapitel 4.8)	Handskizzen, Szenarien, 'wizard of oz' usw. Erstellung von Ablaufplänen usw. mittels (semi-)formaler Methoden	visuelle + verbale Interpretation visuelle + verbale Interpretation bei entsprechender Qualifikation	Spezifikation der Ein-/Ausgabe-schnittstelle (semi-)formale Beschreibungs-Dokumente	Minuten – Tage Stunden – Wochen
Prototyping (siehe Teil B, Kapitel 4.10)	horizontales Prototyping partiell vertikales Prototyping vollständig vertikales Prototyping	'lautes Denken', 'walk-through' heuristische Evaluation aufgabenorientierte Benutzungs-, Usability-Tests	Spezifikation der Dialogkomponente Spezifikation von Teilen der Anwendungskomponente Spezifikation der Anwendungskomponente	Tage – Wochen Wochen – Monate Wochen – Monate
Versioning	erster Durchlauf des gesamten Entwicklungszyklus nächster Durchlauf des gesamten Entwicklungszyklus	induktive Benutzungs-, Usability-Tests deduktive Benutzungs, Usability-Tests	erste weitgehend vollständige Version mehrere weitgehend vollständige Versionen	Monate – Jahre Monate – Jahre

Unter dem *divisionalen Synchronisationsprinzip* verstehen wir die Aufteilung des insgesamt zu erstellenden Funktionsumfanges des Anwendungs-

systems in relativ unabhängige, produktorientierte Bereiche. Durch diese Aufteilung wird der Verkopplungsgrad zwischen den verschiedenen Optimierungszyklen minimiert. Je unabhängiger im Gesamtprojekt die einzelnen produktorientierten Bereiche in der Organisationsgestaltung konzipiert werden, desto wichtiger ist es, zu Beginn des Projektes eine möglichst optimale Aufteilung zwischen den Funktionsbereichen anzustreben. Hieran kann man erkennen, dass Softwaregestaltung sehr eng an Arbeits- und Organisationsgestaltung gekoppelt ist. Da jedoch die Trennung der einzelnen Funktionsbereiche des Anwendungssystems niemals in vollständig unabhängige Bereiche erfolgen kann, müssen weitere Synchronisationsprinzipien zum Einsatz kommen.

Durch das *informationelle Synchronisationsprinzip* wird eine informationelle Kopplung zwischen den einzelnen Optimierungszyklen aufgebaut, sodass alle Personen in den unterschiedlichen Zyklen über den aktuellen Stand in den parallel aktiven Zyklen informiert sind. Dies kann durch einfache Hilfsmittel wie Aktenordner an einem definierten Standort, durch regelmässige Besprechungstermine, aber auch mittels technischer Unterstützung (Mailbox, Versionendatenbanken, 'information repository' usw.) realisiert werden.

Das *personelle Synchronisationsprinzip* besteht darin, dass der Personenkreis, welcher an den unterschiedlichen Optimierungszyklen teilnimmt, *derselbe* ist. Ein Projektmitglied, welches z.B. an einer weiteren Sitzung zur Anforderungsanalyse teilnimmt, kann nicht *gleichzeitig* programmieren; hierdurch wird gewährleistet, dass alle oder zumindest ein wesentlicher Teil der Projektmitglieder den aktuellen Wissensstand haben. Dieses Prinzip bricht sich jedoch oftmals an der arbeitsteilig orientierten Organisationsform, welche bei vielen Softwarehäusern noch vorhanden ist. Hier müsste eine Reorganisation dieser Softwareentwicklungsabteilungen vorgenommen werden.

In diesem Kapitel werden verschiedene Methoden zur Benutzerbeteiligung aufgeführt, welche sich an den verschiedenen Stellen des iterativ-zyklischen Ablaufmodells einsetzen lassen. Die Befragungsmethoden sind z.B. so generell einsetzbar, dass sie in verschiedenen Quadranten zum Einsatz kommen können. Die Benutzungs- bzw. Usability-Tests werden ausführlicher als die anderen Methoden dargestellt, weil sie bisher im deutschsprachigen Bereich noch nicht weiter in dieser Form dargestellt worden sind (siehe dazu speziell Nielsen 1993 sowie Dumas und Redish 1993). Die folgende Tabelle 6 kann dabei behilflich sein, die Zuordnung einzelner – im

folgenden genauer erläuterter – Methoden zu den verschiedenen Projektaktivitäten (Analyse, Bewertung, Darstellung, Produktion, Test) vornehmen zu können (weitere Methoden sind in Heilmann 1981, Wicke 1988, Greenbaum und Kyng 1991 sowie Haberfellner et al. 1992 beschrieben). Wir haben an verschiedenen Stellen in diesem Buch dargelegt, dass sich produktive Methoden besonders gut auch und gerade in der Analysephase eignen, um die gemeinsamen Vorstellungen in bezug auf die wichtigen und notwendigen Anforderungen herausarbeiten zu können.

	A	**B**	**D**	**P**	**T**
Arbeitsanalyse mit Raster bzw. standardisierten Erhebungsverfahren (siehe Übersicht 3 im Anhang)	X	X			
Benutzerorientierte Benutzungstests		X			X
Beobachtung (strukturiert), Videoaufnahme	X				X
Comic strips, Paper pencil		X	X	X	
Diagramme (Flowchart, HIPO usw.)	X		X	X	
Dokumentenanalyse (z.B. Stellenbeschreibung)	X	X			
Fragebogen (z.B. Registrierkarte mit Fragebogen)	X	X		X	
Generatoren, Tools (z.B. Maskengenerator)				X	
Interview, Gruppendiskussion	X	X		X	X
Kärtchenmethode (z.B. Metaplan)	X			X	
Kreativitätstechniken (z.B. Brainstorming)				X	
Logfile (Datenschutz!)	X	X			X
Prototyping	X	X	X	X	X
Review, Walk-through		X			X
Szenarien			X	X	X
Workshop	X	X		X	X

4.2 KONZEPTION EINES BETEILIGUNGSMODELLS

Das hier beschriebene Modell der Benutzerbeteiligung setzt nach der Konzeption des Arbeitssystems im ersten Schritt zur Softwareentwicklung ein. Es steht also bereits fest,

(1) wie die organisatorische Struktur des Arbeitssystems aussieht,

(2) welche Aufgaben in diesem Arbeitssystem bearbeitet werden,

(3) welche Aufgaben mit Hilfe des Softwaresystems unterstützt werden sollen.

Bei der Softwareentwicklung im engeren Sinne wird es nun primär um die Frage gehen, wie die Aufgaben am besten mit Software zu unterstützen sind. Unser Modell der Softwareentwicklung integriert für die Bearbeitung dieser Frage die zukünftigen Benutzer in den Softwareentwicklungsprozess. Der Erfolg dieses Vorhabens hängt dabei entscheidend von der Transparenz ab, mit welcher die Benutzerbeteiligung durchgeführt wird. Es kann immer wieder beobachtet werden, dass die Benutzer in Softwareentwicklungsprojekten nicht aktiv mitarbeiten wollen, weil die Beteiligung erst beim Auftreten von Problemen im Entwicklungsprozess ad hoc organisiert wird. Oftmals fehlt den Benutzern in diesen Situationen der Überblick über die Gesamtzusammenhänge im Projekt, stellt sich bei ihnen das – nicht sehr motivierende – Gefühl ein, plötzlich bei der Lösung von Problemsituationen wichtig zu sein, ohne zu deren Entstehung beigetragen zu haben.

Um das Auftreten solcher Schwierigkeiten möglichst zu verhindern, sollte vor Beginn der eigentlichen Projektarbeiten ein Beteiligungsmodell konzipiert werden, welches als Richtlinie für die Benutzerbeteiligung gilt. So wird die Benutzerbeteiligung für alle Betroffenen durchschaubar und voraussehbar, man kann sich darauf einstellen und Einfluss geltend machen. In diesem Kapitel wird deshalb auf der Grundlage der Ausführungen im Teil A dargestellt, wie ein Beteiligungsmodell erstellt werden kann und welche Kernelemente darin berücksichtigt sein müssen.

Ein Beteiligungsmodell ist das Produkt enger planerischer Kooperation zwischen dem technischen Projektleiter, dem Leiter der direkt betroffenen Fachabteilung, Repräsentanten der indirekt betroffenen inner- und ausserbetrieblichen Instanzen und einem – vor allem bei grösseren Projekten empfehlenswert – eigens für die Beteiligungsorganisation Verantwortlichen. In dieser Phase ist auch die Mitarbeit der Arbeitnehmervertretung von entscheidender Bedeutung. Bereits bei der Erstellung des Beteiligungsmodells

ist also die Zusammenarbeit zwischen Vertretern unterschiedlicher Kompetenzbereiche unabdingbar.

Titelblatt	**Beteiligungsmodell** • Projektname • Verfasser • Verteiler • Datum und Version
1. Teil	*Grundlagen und Voraussetzungen* • Merkmale des Arbeitssystems • Zielsetzung und Ablaufstruktur des Projektes • Projektrahmenbedingungen • Technische Rahmenbedingungen • Personelle Besetzung • Zeitrahmen • Betroffenenstruktur • Endbenutzer • Indirekt Betroffene
2. Teil	*Konkretisierung der Beteiligung in den Projektphasen* • Analyse und Bewertung • Spezifikation • Realisierung • Implementation und Benutzung
3. Teil	*Allgemeine Bestimmungen* • Freistellung • Qualifizierungsmassnahmen für Beteiligung • Informationspolitik für Nichtbeteiligte

Das Beteiligungsmodell wird in einem Dokument festgehalten, welches vor Beginn der eigentlichen Entwicklungsarbeiten allen potentiell Betroffenen zugänglich gemacht werden sollte. Es muss deshalb in allgemein verständlicher, nichttechnischer Sprache abgefasst werden und darf nicht zu umfangreich sein. Es sollte drei Teile umfassen:

1. Teil: Grundlagen und Voraussetzungen

2. Teil: Konkretisierung der Beteiligung in den Projektphasen

3. Teil: Allgemeine Bestimmungen

Tabelle 7 gibt einen Überblick über die inhaltliche Grobstruktur des Beteiligungsmodells. Bevor nun die einzelnen Teile des Beteiligungsmodells näher beschrieben werden, müssen zwei Anmerkungen vorangestellt werden:

(1) Es ist schwierig, wenn nicht sogar unmöglich, Softwareentwicklungsprozesse im voraus bis in die kleinste Einzelheit zu planen. Änderungen in den Anforderungen, technische Pannen, Personalfluktuation und finanzielle Probleme stellen nur einige der Gründe dar. Deshalb wäre es falsch, das Beteiligungsmodell von Beginn weg allzu detailliert auszulegen. Die anschliessend beschriebene Form ist deshalb als ein Rahmen zu betrachten, der im Verlauf des konkreten Entwicklungsprojektes kontinuierlich ausgearbeitet und verfeinert wird.

(2) Trotz der oben angesprochenen Dynamik von Softwareentwicklungsprozessen erhält das Beteiligungsmodell seine Bedeutung nur durch eine gewisse Verbindlichkeit, die ihm von allen beteiligten Interessenvertretungen anerkannt werden muss. Es muss in diesem Sinne durchaus einen Bestandteil der offiziellen Projektabmachungen zwischen den Kooperationspartnern bilden.

4.2.1 Grundlagen und Voraussetzungen

Im ersten Teil des Beteiligungsmodells werden die wichtigsten Grundlagen und Voraussetzungen für das Softwareentwicklungsprojekt so beschrieben, dass alle Betroffenen einen Überblick über die Gründe, Zielsetzungen und Rahmenbedingungen des Projektes erhalten. Im einzelnen sind dies

Merkmale des Arbeitssystems

Als Einstieg in das Beteiligungsmodell werden die Ergebnisse des ersten vorbereitenden Schrittes – Analyse, Bewertung und Gestaltung des Arbeitssystems – dargestellt. Besondere Aufmerksamkeit ist dabei der Beschreibung der zukünftigen Form des Arbeitssystems zu widmen. Damit wird erreicht, dass zwischen allen Beteiligten im Softwareentwicklungsprozess Einigkeit über den Zielzustand des Arbeitssystems herrscht und der Einsatzbereich der zukünftigen Softwareanwendung klar ist.

Zielsetzung und Ablaufstruktur des Projektes

Dieser Teil enthält erste Angaben über allgemeine Merkmale der zukünftigen technischen Unterstützung und eine Übersicht über das vorgesehene Vorgehen bei der Entwicklung. Als allgemeine Merkmale gelten einerseits die Art der Rechnerintegration (z.B. Netzwerk, unabhängige Arbeitsplatzrechner), die Art der Anwendung (z.B. Datenbank, Textverarbeitung, Funktionsintegration) sowie softwareergonomische Kriterien, die von besonderer Relevanz sind (z.B. Transparenz, Flexibilität). In der Übersicht über die Vorgehensschritte bei der Entwicklung werden die Beteiligten über die logische Ablaufstruktur des Projektes informiert.

Projektrahmenbedingungen und technische Rahmenbedingungen

Softwareentwicklungsprojekte können selten auf der 'grünen Wiese' gestartet werden. Deshalb sollte von Beginn weg im Beteiligungsmodell dargelegt werden, ob und gegebenenfalls welche Vorgaben bezüglich Hardware und Betriebssystemen, Einbindung bestehender Softwarelösungen und sonstiger technischer Installationen zu berücksichtigen sind.

Personelle Besetzung

Die Beschreibung der personellen Besetzung des Entwicklungsprojektes gibt den Beteiligten Auskunft über die Aufgabenzuordnung im Softwareentwicklungsteam. Mit diesen Angaben wissen die Beteiligten, mit wem sie es im konkreten Fall zu tun haben, und können mit Fragen, Unklarheiten und Ideen direkt an die entsprechenden Personen gelangen.

Zeitrahmen

Die Kenntnis des zeitlichen Rahmens für die Softwareentwicklung ermöglicht es den Betroffenen, einerseits ihren Einsatz als Beteiligte vorauszuplanen und andererseits realistische Erwartungen für den Zeitpunkt der Inbetriebnahme der Anwendung zu haben. Allfällige Verzögerungen im Projektablauf können anhand dieses Referenzzeitrahmens transparent und verständlich dargestellt werden.

Betroffenenstruktur und Endbenutzer

Damit bei der Durchführung der Benutzerbeteiligung in den verschiedenen Phasen des Projektablaufes die Repräsentativität eingehalten werden kann, lohnt sich die Erstellung einer Übersicht über die Betroffenen. Vor allem

für die Gruppe der zukünftigen Endbenutzer sollte die Übersicht über die folgenden Benutzermerkmale Auskunft geben:

(1) Vorgesehene Aufgaben im neuen Arbeitssystem,

(2) hierarchische Stellung im Betrieb,

(3) Beruf, Ausbildung,

(4) Computererfahrung,

(5) Sprache,

(6) Alter,

(7) Dauer der Betriebszugehörigkeit.

Aufgrund dieser Benutzermerkmale können für die Projektarbeit Benutzergruppen mit jenen Merkmalsausprägungen zusammengestellt werden, die der anstehenden Problemstellung am besten entsprechen. Die Benutzermerkmale können vor allem in grossen und geografisch dezentralen Unternehmen mittels eines einfachen Fragebogens, der von den zukünftigen Benutzern selbst ausgefüllt wird, erhoben werden.

Indirekt Betroffene

Indirekt betroffen sind vor- und nachgelagerte Betriebsbereiche, Kunden und Personen, die selbst nicht mit der zukünftigen Anwendung arbeiten werden, aber von deren Einsatz durch Veränderungen in ihrem eigenen Bereich betroffen sind. Die indirekt Betroffenen sind deshalb vor allem bei Fragen der Schnittstellengestaltung und Systemabgrenzung gegen aussen in den Entwicklungsprozess einzubeziehen. Für die Konzeption der Benutzerbeteiligung genügt in der Regel die Aufzählung der verschiedenen Gruppen, ohne dass detaillierte Merkmalsausprägungen wie bei den Endbenutzern spezifiziert werden müssen.

4.2.2 Konkretisierung der Beteiligung in den Projektphasen

Nach dem Gesamtüberblick im ersten Teil wird im zweiten Teil des Beteiligungsmodells die geplante Benutzerbeteiligung für jede der angegebenen Projektphasen dargestellt. Der Detaillierungsgrad hängt dabei stark von der Projektgrösse ab. Vor allem bei Projekten, welche eine Vielzahl unterschiedlicher Aufgaben und Benutzer betreffen, ist es wichtig, das Beteili-

gungsmodell zu diesem Zeitpunkt noch nicht allzu detailliert zu konzipieren. Folgende Dimensionen sollten aber zumindest grob umrissen werden.

Erarbeitung und Evaluation

Beteiligungsaktivitäten unterteilen sich in solche, bei welchen Lösungen konstruktiv erarbeitet werden, und solche, bei welchen erarbeitete Lösungen geprüft, bewertet und gegebenenfalls überarbeitet werden. Diese Trennung ist vor allem vor dem Hintergrund iterativ-zyklischer Prozessabläufe bedeutsam. Je nach Aufgabe und Struktur der Benutzergruppe kann es sinnvoll sein, für die Erarbeitung und Evaluation unterschiedliche Personen zu beteiligen. Durch die personell getrennte Durchführung können Lösungen auf einer breiteren Basis abgestützt werden.

Direkt – indirekt

Die Unterscheidung zwischen direkter und indirekter Beteiligung bezieht sich auf die Form der Kommunikation und Kooperation zwischen Softwareentwicklern und Beteiligten. Direkte Beteiligung ist gegeben, wenn Kommunikation und Kooperation in unmittelbarer Interaktion vorliegt. Indirekte Beteiligungsformen beschränken sich auf schriftliche Erhebungen, Vertretung der Endbenutzer durch Personen, die im zukünftigen Arbeitssystem nicht selbst die technisch unterstützten Aufgaben wahrnehmen (z.B. Vorgesetzte, Organisatoren usw.).

Inhalt

Der Inhalt der Beteiligung bezieht sich auf den Gegenstand, der bearbeitet wird. In der Phase der Analyse und Bewertung bilden die Datenstruktur, die Aufgabenstrukturen und die Systemfunktionalität der einzelnen zu unterstützenden Aufgaben die zentralen Inhalte. In den übrigen Phasen sind die Ein-/Ausgabeschnittstelle sowie die Dialogschnittstelle von zentralem Interesse. Im Interesse der Klarheit muss aus dem Beteiligungsmodell klar ersichtlich sein, auf welche Aufgaben oder Teilaufgaben sich ein Inhalt bezieht.

Die Informationen zu den Aufgaben 1 und 2 in der Abbildung 13 könnten im Rahmen eines Softwareentwicklungsprojektes für eine Schulverwaltung beispielsweise folgendes bedeuten

Aufgabe 1: Fortlaufendes Erfassen der Prüfungsnoten der Schüler während eines Semesters.

Aufgabe 2: Berechnen des Notendurchschnittes am Semesterende.

Daten: Inhalt ist die vollständige Erfassung und Strukturierung aller mit diesen Aufgaben verbundenen Daten und ihrer Merkmale (Lehrer, Fach, Schülername usw.).

Methode

Für jeden Inhalt wird in Abhängigkeit der Dimensionen 'Erarbeitung - Evaluation' und 'direkt - indirekt' die Methode genannt, die bei der Benutzerbeteiligung angeführt wird. Hier eignen sich zum besseren Verständnis für die Benutzer allgemein formulierte Begriffe wie Workshop, Fragebogen, Interview.

Benutzergruppe

Für jeden Inhalt werden aufgrund der Betroffenenstruktur, wie sie im ersten Teil des Beteiligungsmodells aufgeführt ist, die Benutzergruppen repräsentativ zusammengestellt. Die Merkmalsausprägungen für jede Benutzergruppe (z.B. Sprache, Ausbildung, zukünftige Aufgaben, EDV-Erfahrung) sind so darzustellen, dass sich an der Beteiligung Interessierte selbst einordnen können. Aufgrund dieser Einordnung und der persönlichen Interessen bezüglich Inhalte, Erarbeitung, Evaluation, Methode und zeitlicher Lage besteht somit für jeden Betroffenen die Möglichkeit, sich selbst aktiv für eine Beteiligung am Softwareentwicklungsprozess zu melden. Konkret können die Benutzergruppen wie folgt zusammengesetzt sein:

Gruppe A: Sachbearbeiter der verschiedenen Abteilungen,

Gruppe C: Lehrer,

Gruppe D: Schulleitung, Ebene Abteilungsleitung,

Gruppe F: Schülervertretung.

Abfolge

Die Abfolge gibt das zeitliche Aufeinanderfolgen der einzelnen Aktivitäten an. Damit werden iterativ-zyklische Vorgehensweisen transparent gemacht. Gleichzeitig geplante Aktivitäten erhalten die gleiche Zahl in der Reihenfolge. Die Planbarkeit für die Beteiligten erhöht sich entscheidend, wenn nebst der Reihenfolge auch bereits der Zeitraum angegeben werden kann, für welchen eine Beteiligungsaktivität geplant ist (z.B. Projektmonate 5, 6,

7). Für die Analyse der Datenstruktur der Aufgaben 1 und 2 sieht das Beteiligungsmodell also folgendes Vorgehen vor:

(1) In einem ersten Schritt (Abfolge Nr. 1) werden bei den Sachbearbeitern, Lehrern und den Abteilungsleitern Informationen über die benötigten Daten für Aufgabe 1 und 2 mit einem Fragebogen erhoben. Während der Fragebogenaktion findet ein Workshop statt, in welchem interessierte Sachbearbeiter und Lehrer zusammen mit dem Softwareentwickler erste Überlegungen zur Datenstruktur anstellen.

Ergebnis dieser Aktivitäten ist eine schriftliche Datenliste, in welcher der Softwareentwickler die Daten und Merkmalsausprägungen festhält.

(2) Diese Datenliste wird in einem zweiten Schritt (Abfolge Nr. 2) von den Sachbearbeitern, Lehrern und Abteilungsleitern durchgesehen und korrigiert.

(3) In einem dritten Schritt (Abfolge Nr. 3) erarbeiten nun Sachbearbeiter, Lehrer, Abteilungsleiter und auch ein Vertreter der Schülerschaft zusammen mit dem Softwareentwickler das vollständige und korrekte Datenmodell für die Bearbeitung dieser Aufgaben.

Bei diesem Vorgehen ist es Aufgabe des Softwareentwicklers, die Datenmodelle der einzelnen Aufgaben im Hinblick auf das Gesamtdatenmodell der Softwareanwendung hin zu koordinieren und allfällige Inkompatibilitäten im Verlauf des weiteren Vorgehens zur Diskussion zu stellen.

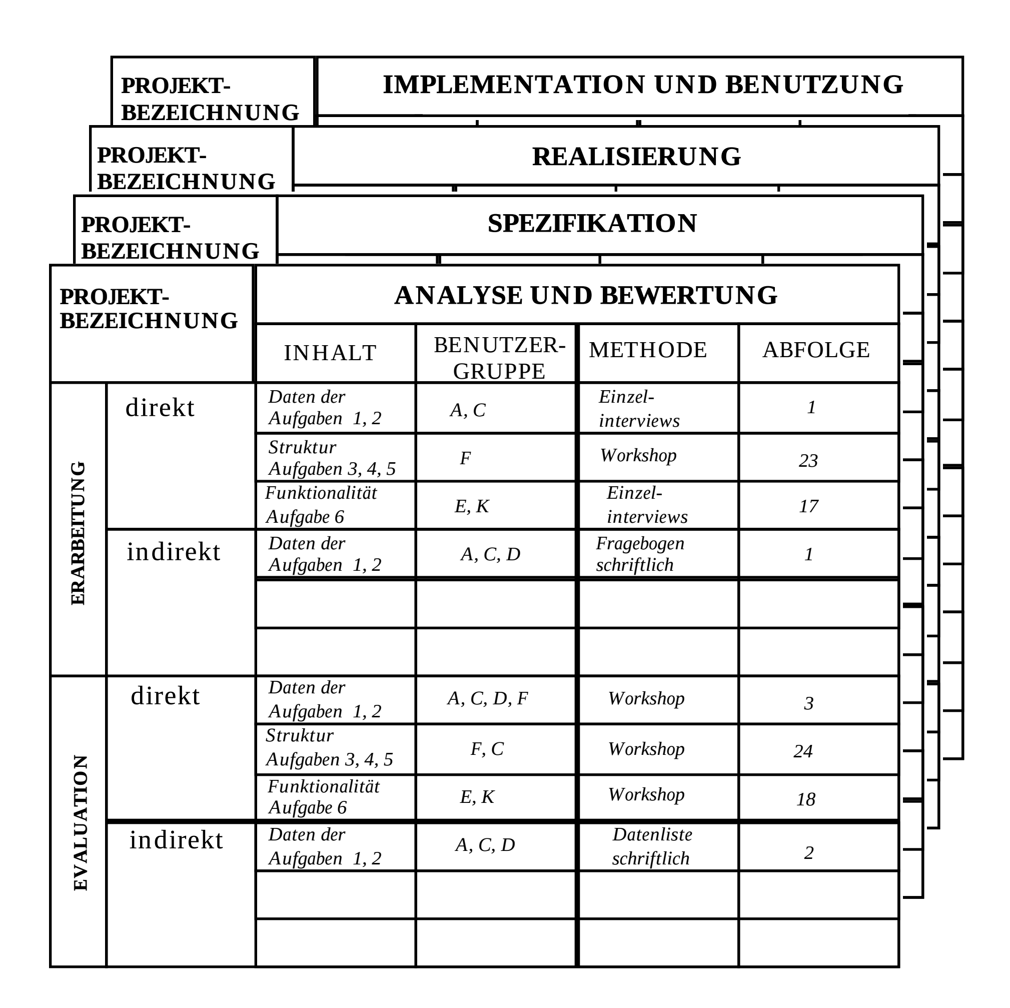
Es ist selbstverständlich, dass die Beteiligung der Endbenutzer gerade in grossen Projekten nicht in diesem Detaillierungsgrad vorausplanbar ist. Das Modell unterstützt jedoch auch die laufende Planung und Kontrolle während der Projektarbeit, weist auf Unterlassungen hin und hilft, die einzelnen Aktivitäten zu koordinieren. So können zum Beispiel in einem einzelnen Workshop mit einer bestimmten Benutzergruppe verschiedenste Inhalte bearbeitet werden. Mit der Anwendung eines solchen Modelles soll erreicht werden, dass

(1) der Beteiligungsprozess – als ganzer – für alle transparent wird;

(2) dadurch die Möglichkeit geboten wird, persönliche Beteiligungswünsche zu formulieren.

Abbildung 13 zeigt, wie die verschiedenen Dimensionen schematisch dargestellt werden können. Die Übernahme der Dimensionen und Informationen in eine Datenbank ermöglicht es zudem, während des Beteiligungspro-

zesses Informationen gezielt zusammenzustellen und abzufragen. So können z.B. sämtliche Beteiligungsaktivitäten zu einem bestimmten Inhalt oder für eine bestimmte Benutzergruppe übersichtlich kontrolliert und kommuniziert werden.



PROJEKT-BEZEICHNUNG — IMPLEMENTATION UND BENUTZUNG

PROJEKT-BEZEICHNUNG — REALISIERUNG

PROJEKT-BEZEICHNUNG — SPEZIFIKATION

PROJEKT-BEZEICHNUNG		ANALYSE UND BEWERTUNG			
		INHALT	BENUTZER-GRUPPE	METHODE	ABFOLGE
ERARBEITUNG	direkt	*Daten der Aufgaben 1, 2*	*A, C*	*Einzel-interviews*	*1*
		Struktur Aufgaben 3, 4, 5	*F*	*Workshop*	*23*
		Funktionalität Aufgabe 6	*E, K*	*Einzel-interviews*	*17*
	indirekt	*Daten der Aufgaben 1, 2*	*A, C, D*	*Fragebogen schriftlich*	*1*
EVALUATION	direkt	*Daten der Aufgaben 1, 2*	*A, C, D, F*	*Workshop*	*3*
		Struktur Aufgaben 3, 4, 5	*F, C*	*Workshop*	*24*
		Funktionalität Aufgabe 6	*E, K*	*Workshop*	*18*
	indirekt	*Daten der Aufgaben 1, 2*	*A, C, D*	*Datenliste schriftlich*	*2*

Abbildung 13: Schematische Auslegung der Benutzerbeteiligung.

4.2.3 Allgemeine Bestimmungen

Der dritte und letzte Teil des Beteiligungsmodells enthält allgemeine Bestimmungen, die eine möglichst konfliktfreie und reibungslose Durchführung der Benutzerbeteiligung unterstützen. Zahlreiche Erfahrungen in den verschiedensten Projekten zeigen immer wieder, dass Beteiligungsprozesse wegen unklarer Abmachungen und Kompetenzabgrenzungen schwierig werden und dadurch die Motivation aller involvierten Stellen stark abnimmt. Wir empfehlen deshalb, primär die folgenden drei Bereiche über klare Abmachungen zu regeln.

Freistellung

Insbesondere bei Formen der direkten Benutzerbeteiligung, bei welchen Mitarbeiter der Fachabteilungen regelmässig für Analysen, Workshops oder über bestimmte Zeiträume hinweg der Softwareentwicklung zur Verfügung stehen müssen, ist geregelte Entlastung vom Alltagsgeschäft im Betrieb unabdingbare Notwendigkeit. Leider versuchen Linienvorgesetzte immer wieder, gerade jene Mitarbeiter freizustellen, auf die wegen fehlender Fachqualifikationen oder kurzer Betriebszugehörigkeit im Alltagsgeschäft der Fachabteilung eher verzichtet werden kann. Diese Personen bringen denn auch oft nicht das notwendige Wissen in den Softwareentwicklungsprozess ein, so dass gute und innovative Lösungen schon bald gefährdet sein können.

Qualifizierungsmassnahmen für die Beteiligung

Die aktive und direkte Mitarbeit in Softwareentwicklungsprozessen setzt sowohl bei den Softwareentwicklern als auch bei den Beteiligten selbst gewisse soziale und fachliche Qualifikationen voraus. Vor allem beim Einstieg in die Softwareentwicklung mit direkter Benutzerbeteiligung ist deshalb darauf zu achten, dass diese Qualifikationen vor Projektbeginn vermittelt werden.

Informationspolitik für Nichtbeteiligte

Oftmals ist es gerade bei Entwicklungsprojekten mit einer grossen Anzahl von Betroffenen unmöglich, dass alle Betroffenen direkt oder indirekt beteiligt werden. Die Nichtbeteiligten sind deshalb regelmässig über Projektverlauf, aktuellen Stand und Entscheidungen zu informieren.

4.3 ANALYSE, BEWERTUNG UND GESTALTUNG BESTEHENDER ARBEITSSYSTEME

Bei der Entwicklung und Einführung eines Softwaresystems sind Fragen der Arbeitsorganisation, der Mensch-Computer-Funktionsteilung, der Softwaregestaltung und der Hardwaregestaltung zu beachten. In diesem Kapitel zur Arbeitsorganisation gehen wir davon aus, dass die Arbeit dem Menschen sinnvolle Tätigkeiten und Möglichkeiten zur persönlichen Entfaltung bietet und aus sich selbst heraus motivierend wirkt. Im wesentlichen können folgende Vorteile erwartet werden, wenn man sich vorgängig zur Softwareentwicklung mit den arbeitsorganisatorischen Aspekten auseinandersetzt:

(1) Der optionale Charakter der neuen Technologien kann adäquat genutzt werden. Eine bestehende Arbeitsorganisation wird nicht unreflektiert übernommen und durch die neuen Technologien zementiert.

(2) Mit der Software wird nicht nur ein isoliertes Problem gelöst, welches eventuell nur Symptom, nicht aber Ursache einer unbefriedigenden Situation darstellt.

(3) Neue Technologien werden nicht allein aufgrund ihrer Verfügbarkeit eingeführt, sondern gezielt auf die Verbesserung der Situation für die Betroffenen und die Effizienz des gesamten Arbeitssystems hin entwickelt.

(4) Es entsteht sowohl für die zukünftigen Benutzer als auch für die Softwareentwickler von Beginn an ein gemeinsames Verständnis für das gesamte Arbeitssystem. Dadurch können Anforderungen besser definiert werden. Unsicherheiten, Missverständnissen und Widerstand kann vorgebeugt werden.

Besonders letzteres muss für die Softwareentwickler von grossem Interesse sein. Es kann nämlich immer wieder beobachtet werden, dass Computerbenutzer letztlich gar nicht wegen der Unzulänglichkeit der Software unzufrieden sind, sondern wegen unbefriedigender Arbeitsstrukturen, welche durch die Software unterstützt werden. Damit wird deutlich, dass die Gestaltung der Arbeitssituation nicht primär eine Aufgabe der Softwareentwicklung sein kann, sondern eine Aufgabe der Arbeits- und Organisationsfachleute zusammen mit den Beschäftigten des Arbeitssystems. Den Informatikfachleuten fällt hierbei jedoch die wichtige Aufgabe zu, den Prozess der Arbeitsstrukturierung mit ihrem technischen Wissen zu unterstützen, in-

dem sie technische Möglichkeiten und auch Grenzen aufzeigen. Nur so kann von Beginn an die Forderung nach soziotechnisch optimierten Arbeitssystemen eingelöst werden. Gleichzeitig erhalten die Informatikspezialisten durch ihre Teilnahme an diesen Gestaltungsprozessen viel Hintergrundwissen zu den Überlegungen der zukünftigen Anwender, können Anforderungen an das Softwaresystem besser nachvollziehen und achten bei der Softwaregestaltung bewusster darauf, arbeitsorganisatorische Optionen nicht unnötig technisch einzuschränken.

Obwohl sich dieses Buch nicht primär mit Fragen der Arbeitsstrukturierung und Organisationsentwicklung befasst, möchten wir an dieser Stelle einige zentrale Aspekte aus diesem Bereich darstellen. Ohne im Detail auf die Methoden und Konzepte der Organisationsentwicklung einzugehen – dafür verweisen wir auf die umfangreiche Fachliteratur –, werden hier prozessbezogene und methodische Merkmale von Veränderungsmassnahmen erläutert.

Organisationsentwicklungen sind grundsätzlich als Veränderungsprozesse zu begreifen, in welchen ein unbefriedigender Ausgangszustand (Ist-Zustand) eines Arbeitssystems zusammen mit den Betroffenen in einen optimalen Zielzustand (Soll-Zustand) überführt werden soll. Da der Zielzustand zu Beginn eines Veränderungsprozesses meist nur in Form allgemeiner Vorgaben durch die Geschäftsleitung bekannt ist – z.B. mehr Effizienz, grössere Flexibilität, bessere Kundenorientierung, höhere Motivation –, besteht bei den Betroffenen meist eine erhebliche Unsicherheit bezüglich der konkreten Ziele und Realisierungsschritte. Um diese Unsicherheiten abzubauen und grösstmögliche Transparenz und Identifikation mit dem anzustrebenden Zielzustand zu erreichen, ist auch hier wie in der Softwareentwicklung selbst die aktive Beteiligung aller Betroffenen eine wichtige Voraussetzung für das Gelingen der Veränderungsprozesse. Für die Erarbeitung von arbeitsorganisatorischen Vorgaben für die Softwareentwicklung sollte man sich deshalb an ein Vorgehen halten, welches allen Betroffenen erlaubt,

(1) die Ausgangssituation gemeinsam zu beurteilen,

(2) an der Zielfindung für die Zukunft mitzuarbeiten und

(3) daraus Anforderungen und Gestaltungsziele für das zukünftige Arbeitssystem abzuleiten.

Ein solches Vorgehen hat notwendigerweise drei Schritte zum Inhalt:

Schritt 1: Analyse der vorliegenden Situation.

Schritt 2: Definition der Anforderungen an das zukünftige Arbeitssystem.

Schritt 3: Konzeption des neuen Arbeitssystems.

4.3.1 Schritt 1: Analyse der vorliegenden Situation

Ziel:

Die Analyse der vorliegenden Situation bedeutet eine möglichst umfassende Betrachtung der von einer technologischen Innovation betroffenen Organisationseinheit. Die Herausarbeitung der Stark- und Schwachstellen im bestehenden System steht dabei im Vordergrund. Die Analyseergebnisse sollen Aussagen zur Umweltsituation und zur Aufbau- und Ablauforganisation der Organisationseinheit sowie zum technischen und sozialen Teilsystem des Arbeitssystems enthalten (siehe Übersicht 3 im Anhang).

Methodische Zugänge:

Den wohl umfassendsten Zugang zur Analyse der aktuellen Situation bietet die soziotechnische Systemanalyse. Sie berücksichtigt alle in der Zielstellung genannten Dimensionen und erfasst sie in mehreren aufbauenden Analyseschritten. Ausgehend von einer Grobanalyse des Arbeitssystems, in welcher die formale Organisationsstruktur, die wichtigsten Input- und Outputgrössen, deren Transformationsprozesse sowie ökonomische und soziale Ziele beschrieben werden, werden die folgenden zentralen Fragen zu beantworten gesucht.

Fragen zur Umweltsituation:

- Welche Produkte werden heute und in Zukunft vom Markt verlangt?
- Welche Anforderungen stellt die Umwelt an den Produktionsprozess bezüglich Termine, Qualität, Flexibilität usw.?
- Wie sieht die Konkurrenzsituation aus?

Fragen zu Auftrag und Abhängigkeiten:

- Wie sieht der Gesamtauftrag des Arbeitssystems aus?
- In welchen Beziehungen und Abhängigkeiten zu vor- und nachgelagerten Bereichen befindet sich das Arbeitssystem?

- In welche Einzelaufgaben ist der Gesamtauftrag unterteilt und wie sind diese auf die einzelnen Mitarbeiter verteilt?

Fragen zum technischen Teilsystem:

- Mit welchen technischen Betriebsmitteln und Arbeitsverfahren werden die einzelnen Aufgaben bearbeitet?
- Wo und wie oft treten technisch bedingte Störungen und Schwankungen in Form von Ausfällen, Engpässen usw. auf?
- Welche räumlichen Bedingungen sind gegeben?

Fragen zum sozialen Teilsystem:

- Wie sieht die Mitarbeiterstruktur bezüglich Hierarchie, Ausbildung, Betriebszugehörigkeit, Geschlecht usw. aus?
- Wie sieht die Kooperationsstruktur aus und welchen Zusammenhang hat diese mit dem Auftreten und der Art des Umgangs mit Störungen und Schwankungen im Arbeitsablauf?
- Welche Bedürfnisse bezüglich der Arbeit selbst (Verantwortung, Selbständigkeit, Kooperation und Kommunikation, Weiterentwicklung usw.) und bezüglich der Arbeitsbedingungen (Arbeitszeit, Lohn, Mitsprache usw.) haben die Beschäftigten?

Für die Beantwortung zahlreicher dieser Fragen stellt die Arbeitspsychologie spezielle getestete Instrumente und Verfahren zur Verfügung. Nebst den objektiven Verfahren, mit deren Hilfe Experten z.B. Auftrags- und Bedingungsanalysen, Tätigkeitsanalysen und -bewertungen sowie Anforderungs- und Belastungsanalysen durchführen, erscheinen uns vor allem jene Verfahren relevant, mit deren Hilfe die Beschäftigten ihre Arbeitssituation subjektiv analysieren und bewerten können. Diese Verfahren eignen sich in besonderer Weise für die Mitarbeiterbeteiligung in Veränderungsprozessen. Ihre Ergebnisse können ohne allzu grossen Auswertungsaufwand und gut verständlich zurückgemeldet werden. So bilden sie eine wichtige Grundlage für die Definition von Anforderungen an das neue Arbeitssystem im nächsten Schritt.

4.3.2 Schritt 2: Definition der Anforderungen an das zukünftige Arbeitssystem

Ziel:

Die Ergebnisse der Situationsanalyse und vor allem die Beschäftigung mit den Stärken und Schwächen erlauben es nun, Anforderungen an die zukünftige Situation zu erstellen. Die Anforderungen sollen dabei so formuliert werden, dass daraus konkrete Gestaltungsziele abgeleitet werden können. Anforderungen müssen sich dabei sowohl auf die Bedürfnisse der Organisation wie auch auf die Bedürfnisse der Beschäftigten beziehen.

Methodische Zugänge:

Für die Definition von Anforderungen eignen sich am besten gruppenorientierte Vorgehensweisen, bei welchen Kreativitätstechniken wie Brainstorming oder Metaplan ideal eingesetzt werden können. Bei der Definition von Anforderungen ist es von äusserster Wichtigkeit, dass eventuelle Änderungen am Gesamtauftrag des Arbeitssystems allen Beteiligten klar sind und von ihnen auch unterstützt werden. Zudem ist es gruppendynamisch wichtig, dass das Arbeiten in Gruppen von einem Moderator begleitet wird und dass man sich an vorher gemeinsam aufgestellte 'Spielregeln' hält. Da die Gruppen idealerweise aus Personen mit unterschiedlichen betrieblichen Stellungen zusammengesetzt sind, ist häufig eine Moderation durch einen betriebsexternen Moderator angebracht.

Damit bei der Anforderungsdefinition die ganzheitliche Sicht aus der Situationsanalyse nicht verlorengeht – und dadurch nur Anforderungen zur Lösung isolierter Probleme genannt werden –, lohnt es sich, zu Beginn themenorientiert vorzugehen. Eine mögliche Themengliederung ergibt sich schon aus der Gliederung der Situationsanalyse:

(1) Anforderungen aus den Schwachstellen der Umweltbeziehungen,

(2) Anforderungen aus den Schwachstellen der Abhängigkeiten und dem Arbeitsablauf,

(3) Anforderungen aus den Schwachstellen im technischen Teilsystem,

(4) Anforderungen aus den Schwachstellen im sozialen Teilsystem.

Zu jedem Themenblock liegen aus der Situationsanalyse konkrete Ergebnisse vor, aufgrund deren nun spezifische Anforderungen formuliert werden können. In die Formulierung der Anforderungen sollten jedoch noch keine konkreten Lösungsvorschläge einfliessen.

In einem nächsten Schritt werden die genannten Anforderungen aufeinander bezogen, nach organisations- und beschäftigtenspezifischen Kriterien kategorisiert und – dies ist besonders wichtig – gewichtet. Unterschiedliche Gewichtungen einzelner Anforderungen durch verschiedene Beteiligte weisen auf unterschiedliche Interessenlagen hin. Die Gewichtung ermöglicht es, verschiedene Standpunkte explizit zu machen und Lösungen auszuhandeln.

Anforderungen sollten immer möglichst positiv formuliert werden, z.B.:

Wir brauchen ...
... eine Verringerung des administrativen Aufwandes
... direktere Kommunikationswege
... grössere Aktualität der Informationen
... grössere Nähe zum Kunden

Wir wollen ...
... mehr Verantwortung für die eigenen Aufgaben
... bessere Nutzung der vorhandenen Qualifikationen
... Schaffung neuer Qualifikationsmöglichkeiten
... mehr Abwechslung in der Arbeit

4.3.3 Schritt 3: Konzeption des neuen Arbeitssystems

Ziel:

Bei der Konzeption eines neuen Arbeitssystems werden nun für die in Schritt 2 erstellten Anforderungen konkrete Gestaltungsziele und Lösungen erarbeitet. Die optimale Lösung soll in der anschliessenden Softwareentwicklung als verbindliche Vorgabe für die Gestaltung der technischen Unterstützung dienen.

Methodische Ansätze:

Generell gelten auch bei diesem Schritt jene Moderationsmethoden als geeignet, die viel Platz für Kreativität und Aushandlungsprozesse in Gruppen offenlassen. Da konkrete Gestaltungslösungen immer in mehreren Varianten erstellt werden sollten, lohnt sich deren Erarbeitung in parallelen Arbeitsgruppen. Die einzelnen Lösungsvarianten können dann anhand der Anforderungen aus Schritt 2 evaluiert und gemeinsam zu einer optimalen Variante integriert werden. Damit das Ergebnis der Konzeption des neuen Arbeitssystems auch wirklich als praktikable Leitlinie in der Softwareentwicklung zum Tragen kommt, sollen folgende Fragen beantwortet sein:

(1) Wie sieht die Organisationsstruktur im neuen Arbeitssystem aus?

(2) Wie werden der Arbeitsablauf und die einzelnen Aufgaben gestaltet?

(3) Welches sind die Grundmerkmale der technischen Unterstützung?

Gerade für die Beantwortung der letzten Frage ist die Mitarbeit jener Softwareentwickler, die anschliessend die technische Unterstützung für das neue Arbeitssystem konzipieren und realisieren, unumgänglich. Ihr Beitrag darf jedoch nicht darin bestehen, für die Mitarbeiter gute Lösungsvarianten durch das Anführen technischer Sachzwänge präventiv zu unterbinden. Ihr Auftrag besteht vielmehr darin, im Rahmen der technischen Möglichkeiten mit den zukünftigen Benutzern zusammen jene technische Lösung zu finden, die die gewünschte Form des Arbeitssystems optimal unterstützt.

Zum Schluss dieser Ausführungen wollen wir noch kurz einige Anmerkungen zur Situation bei den Projekttypen C und D (Branchen- und Standardsoftware) anführen. Diese Projekttypen zeichnen sich ja gerade dadurch aus, dass dabei die Software nicht für ein bestimmtes Arbeitssystem entwickelt wird, sondern für eine Klasse von Arbeitssystemen, welche die in der Software enthaltene Funktionalität nutzen können. Auch bei diesen erachten wir es als möglich und notwendig, bei verschiedenen möglichen Kunden Abklärungen zur Arbeitsorganisation vorzunehmen. Zu oft werden diese Systeme nämlich ausschliesslich auf ihre Funktionalität hin entwickelt oder werden aufgrund einer Individuallösung für einen spezifischen Kunden dann in der Branche breit angeboten. Organisatorische Kontexte des einzelnen Anwenders fallen so in der Entwicklung ausser Betracht und zwingen den Anwender oftmals zu ungewollten Anpassungen seiner Arbeitsstrukturen an die Technik. Mit Organisationsanalysen in diversen potentiellen Anwenderbetrieben wird es möglich, prospektiv den Zugang zur Systemfunktionalität derart zu gestalten, dass ein Softwaresystem in unterschiedlichen Organisationsstrukturen flexibel einsetzbar wird.

4.4 Vorgehen zur Ermittlung erster Anforderungen an die Software

Die Ermittlung der vollständigen und richtigen Anforderungen an eine zukünftige Anwendungssoftware stellt trotz der Fortschritte bei der Werkzeugunterstützung für die Softwareentwicklung weiterhin ein äusserst schwieriges Problemfeld dar.

Es können vier Arten von aufgabenbezogenen Anforderungen unterschieden werden:

(1) Die erste Art bezieht sich auf Anforderungen, die sich aus der Aufgabengestaltung ergeben. Software muss so gestaltet sein, dass sie Möglichkeiten für Aufgabenorientierung in der Arbeit, wie sie im Teil A, Kapitel 1.2 beschrieben ist, schafft oder zumindest nicht beschneidet.

(2) Die zweite Art von Anforderungen ergibt sich aus der Beziehung zwischen Aufgabe, Benutzer und Computer. Sie regeln die Funktionsteilung zwischen Mensch und Maschine so, dass die spezifischen Stärken des jeweiligen Funktionsträgers optimal genutzt werden.

(3) Anforderungen einer dritten Art beziehen sich auf die Benutzungsoberfläche. Sie bestimmen also, wie ein Benutzer auf die implementierte Systemfunktionalität zugreift und wie der Zugriff auf die im System enthaltenen Informationen im Rahmen der Arbeitsaufgabe zu geschehen hat.

(4) Die vierte Art von Anforderungen ergibt sich aus dem Aufgabengegenstand und meint die Vollständigkeit und richtige Beschreibung der zur Aufgabenerfüllung benötigten Daten und deren Beziehungen untereinander.

Betrachtet man die in der heutigen Praxis vorwiegend angewandten Vorgehensweisen und Methoden bei der Anforderungsanalyse und Systemdefinition, so fällt die Konzentration auf die Anforderungen der vierten Art (Daten und deren Beziehungen) auf. Im Hinblick auf die Erarbeitung der technischen Anforderungen für das Funktions- und Datenmodell einer Anwendung erscheint dies auch sinnvoll und notwendig. Geht man davon aus, dass die ersten zwei Arten von Anforderungen (Aufgabenorientierung und Mensch-Computer-Funktionsteilung) bereits bei der Auslegung des Arbeitssystems im Vorfeld des Softwareentwicklungsprozesses umrissen wurden, so wird es im Hinblick auf die Aufgabenangemessenheit einer Software unter dem Aspekt der zunehmenden Anwendungskomplexität immer wichtiger, bereits auch bei der Anforderungsermittlung software-ergonomische Aspekte in der Analysephase mit zu berücksichtigen. Die softwareergonomische Gestaltung darf sich also nicht mehr darauf beschränken, allgemein abgefasste Gestaltungsrichtlinien in der Realisierungsphase ohne Bezug zur Arbeitsaufgabe umzusetzen und anschliessend aufgrund rein evaluativer Verfahren wie Experimente und Benutzerbefragungen zu korrigieren. Aufgabenangemessenheit in der Softwareentwick-

lung heisst, in der Analysephase prospektiv Erkenntnisse zu gewinnen, die Hinweise auf die softwareergonomische Gestaltung im konkreten Aufgabenkontext des Benutzers geben.

Die folgenden Ausführungen zeigen die wichtigsten Aspekte auf, die für die Bearbeitung softwareergonomischer Fragestellungen bei der Anforderungsermittlung im Softwareentwicklungsprozess zu beachten sind. Damit als Ergebnis der Anforderungsermittlung konkrete Gestaltungshinweise für den Systementwurf beim Prototyping gegeben werden können, empfiehlt sich grundsätzlich ein Vorgehen in drei Schritten, die iterativ-zyklisch durchlaufen werden können.

4.4.1 Schritt 1: Analyse der Struktur und Daten von Arbeitsaufgaben

Beim Entwurf des Arbeitssystems als Vorbereitung zum Softwareentwicklungsprozess wurden die einzelnen Aufgaben festgelegt und umschrieben. Damit für die einzelnen Aufgaben nun die optimale technische Unterstützung gefunden weden kann, müssen für jede Aufgabe getrennt die spezifischen Merkmale bezüglich Strukturen und die Datenstruktur des gesamten Systems analysiert werden. Eine besondere Schwierigkeit bietet hierbei oftmals die Tatsache, dass die zu analysierenden Aufgaben in der geplanten Form noch gar nicht bestehen. *Aufgabenstrukturen und Datenstruktur können also nicht direkt beobachtet oder erfragt werden*, sondern sind in einem konstruktiven Prozess zusammen mit den zukünftigen Benutzern und Betroffenen des Systems zu erarbeiten.

Arbeitsaufgaben können anhand sehr unterschiedlicher Merkmale analysiert und beschrieben werden. So stellen etwa Beck und Ziegler (1991) in einem Projekt zur Technik der aufgaben- und benutzerangemessenen Softwarekonstruktion (TASK) ein Modell mit fünfzehn Aufgabenmerkmalen (Name, Art, Struktur, auslösendes Ereignis, Vor- und Nachbedingung, Häufigkeit, Wiederholungsrate, Priorität, Dauer, Vollständigkeit, Eingriffs- und Auswahlmöglichkeiten, Belastungsfaktoren, Restriktionen, technische Durchführbarkeit, Sicherheit, Komplexität) und mit insgesamt über dreissig Beschreibungskategorien dar, mit dem die relevanten Aspekte einer Arbeitsaufgabe für die Softwareentwicklung vollständig zu erfassen sind. Obwohl wir diese Vollständigkeit als richtig und auch notwendig erachten, erscheint uns als Einstieg in die Aufgabenanalyse mit direkter Benutzerbeteiligung eine Konzentration auf die zentralen Merkmale 'Aufgabenstruktur' und 'Variationen in der Abfolge der Teiltätigkeiten' als sinnvoll.

Die Aufgabenstruktur gibt Auskunft über die Teiltätigkeiten, aus welchen eine Aufgabe zusammengesetzt ist, und die Abfolge bzw. Hierarchie der Teiltätigkeiten. Mit den 'Variationen in der Abfolge der Teiltätigkeiten' werden mögliche alternative Abfolgen aufgezeigt, die je nach Arbeitssituation und persönlichem Arbeitsstil genutzt werden können.

Neben der Analyse der Aufgabenstruktur der einzelnen Aufgaben bildet die Analyse der Daten und die Erstellung einer ersten Datenstruktur und -spezifikation für das gesamte Arbeitssystem den zweiten Schwerpunkt des ersten Vorgehensschrittes. Ziel der Datenanalyse ist es, sämtliche für die Bearbeitung der Arbeitsaufgaben benötigten Daten zu erfassen, ihre Merkmalsausprägungen (Wertebereich, Identifikation usw.) so zu beschreiben, dass sie den softwaretechnischen Kriterien genügen und ihre formalen Beziehungen untereinander darstellen. Für die Analyse der Datenstruktur können diejenigen softwaretechnischen Methoden und Vorgehensweisen eingesetzt werden, die der zukünftigen Anwendung und der vorhandenen Entwicklungsumgebung am besten entsprechen.

4.4.2 Schritt 2: Integration der Analyseergebnisse

Ist die Analyse der Aufgabenstruktur der einzelnen Arbeitsaufgaben abgeschlossen und besteht eine erste Sicht der Datenstruktur und -beschreibung, so werden im zweiten Schritt der Anforderungsermittlung die Ergebnisse des ersten Schrittes derart integriert, dass jeder Aufgabe die dazugehörenden Daten mit ihren spezifischen Merkmalsausprägungen zugeordnet werden. Als spezifische Merkmalsausprägungen gelten zum Beispiel bestimmte Werte aus dem gesamten Wertebereich, die für diese bestimmte Aufgabe zugelassen sind. Mit diesem Vorgehen kann nun festgelegt werden, an welchen Stellen im gesamten Aufgabenkontext ein Datum in welcher Form gebraucht wird, welchen aufgabenspezifischen Bedingungen es jeweils genügen muss, wo es erfasst und geändert werden kann oder wo es reinen Informationszwecken dient.

4.4.3 Schritt 3: Ableitung softwareergonomischer Anforderungen

Damit aufgrund der vorliegenden Informationen zum Zusammenhang zwischen Aufgabenstruktur und Daten erste Anforderungen für den anschliessenden Entwurf abgeleitet werden können, müssen die einzelnen Teiltätigkeiten der Aufgaben noch bezüglich ihres Schwierigkeitsgrades oder nach der zur Bewältigung der Aufgaben erforderlichen kognitiven Anforderung

klassifiziert werden. Ein mögliches Klassifikationsschema für kognitive Anforderungen ist etwa die Unterscheidung der Teiltätigkeiten in solche der reinen Informationsübertragung (die kognitive Anforderung beschränkt sich aufs Wahrnehmen und kurzzeitige Behalten von Informationen), der Informationsverarbeitung (Informationen werden nach vorgegebenen Regeln zugeordnet oder ausgewählt), der Informationsbearbeitung (aus bestehenden Informationen werden nach vorgegebenen Regeln durch Beurteilen, Ordnen und Verknüpfen neue Informationen geschaffen) oder der Informationserarbeitung (neue Informationen werden ohne vorgegebene Regeln erarbeitet). Diese Aufzählung von Klassifikationsmöglichkeiten macht deutlich, dass innerhalb einer einzelnen Aufgabe Teiltätigkeiten mit verschiedenen Anforderungshöhen auftreten können. Mit der Differenzierung innerhalb einer Aufgabe wird es nun möglich, für die einzelnen Teilschritte bereits im Vorfeld zum Prototyping gewisse Gestaltungsregeln der Softwareergonomie als Anforderungen an das technische System festzuhalten, also z.B. zu bestimmen, ob eine bestimmte Teiltätigkeit besser mit einer Informationsdarstellung in Listenform oder in grafischer Form dargestellt wird. Mit der Benennung solcher Anforderungen vor dem Prototyping können auch gezielt Evaluationskriterien aufgestellt werden, die bei der Überprüfung von Prototypen als Orientierungsrahmen verwendet werden können.

Wir werden im folgenden die meisten bekannten Methoden einzeln darstellen, mit denen sich Benutzer an verschiedenen Stellen des Entwicklungsprozesses beteiligen lassen.

4.5 DER WORKSHOP

Neben mündlichen und schriftlichen Umfragen sind Workshops wohl die am meisten genutzte Möglichkeit zum Einbezug von Benutzern; diese Methode eignet sich sehr gut für die Zusammenarbeit von Entwicklern und Benutzern in verschiedenen Stadien des Entwicklungsprozesses, sei es zur Analyse grundlegender Probleme, zur Bewertung verschiedener Lösungsvarianten oder zur Gestaltung der Benutzungsoberfläche. Unter der Bezeichnung 'Joint Application Design' hat sich in Nordamerika in manchen Firmen die *gezielt in den Entwicklungsprozess eingebaute Durchführung mehrerer, stark strukturierter Benutzer-Entwickler-Workshops eingebürgert* (siehe dazu Wood und Silver 1989 sowie August 1991).

Workshops sind eine ausgezeichnete Möglichkeit für die *direkte* Zusammenarbeit von Entwicklern und Benutzern und nutzen die Vorteile der

Gruppendynamik beim Problemlösen. Sofern Workshops nicht schon standardmässig im Entwicklungsprozess vorgesehen sind, können unterschiedliche Ereignisse den Anlass für die Durchführung eines Workshops bilden: z.B. Ergebnisse einer Benutzerumfrage, ein Meilenstein im Projekt, der Wunsch des Projektleiters, offene Fragen mit Benutzern direkt zu klären, usw. Für einen produktiven, erfolgreichen und für alle Beteiligten gewinnbringenden Workshop ist eine gute *Vorbereitung*, eine *Aufbereitung* der Ergebnisse sowie oftmals eine *Moderatorenschulung notwendig.*

4.5.1 Vorbereitung

Es ist unbedingt erforderlich, ein Konzept des Workshops zu erarbeiten, das heisst, Thema, Ziele, Ablauf, Zeitplan, Teilnehmer, Moderation, Fragen, Aufgaben, Methoden, Material, Geräte (z.B. Prototypingwerkzeuge), Räume, Schulungsnotwendigkeiten (z.B. für bestimmte Kreativitätstechniken) zu klären! Für eine detailliertere Darstellung siehe Wood und Silver (1989) sowie August (1991).

Als Teilnehmer sollten für den Anwendungsbereich repräsentative, qualifizierte und interessierte Benutzer, Mitglieder der Projektgruppe (Entwickler) sowie evtl. Spezialisten für bestimmte Fragen (z.B. Werkzeuge, Ergonomie) ausgewählt werden. Die Anzahl der Teilnehmer sollte zehn bis zwölf Personen nicht übersteigen; bei grösserer Teilnehmerzahl ist die gezielte Bildung von Arbeitsgruppen – mit evtl. unterschiedlichen Aufgaben – empfehlenswert. Die Ergebnisse können dann im Plenum zusammengetragen werden. Eine gute Vorbereitung der Teilnehmer durch intensives Studium im voraus abgegebener Unterlagen verkürzt die Einstiegszeit!

4.5.2 Ablauf

Ein möglicher grober *Ablauf* eines Workshops sieht wie folgt aus:

(1) 'Spielregeln' erläutern (siehe Checkliste 2 im Anhang).

(2) Einleitung: Ziel und Zweck; eventuell Präsentation von Umfrageergebnissen, Stand der Dinge.

(3) Rahmenbedingungen abklären.

(4) Prioritäten setzen.

(5) Input (Thema umreissen, erste Frage) durch Moderator:
• Diskussion, Visualisierung;

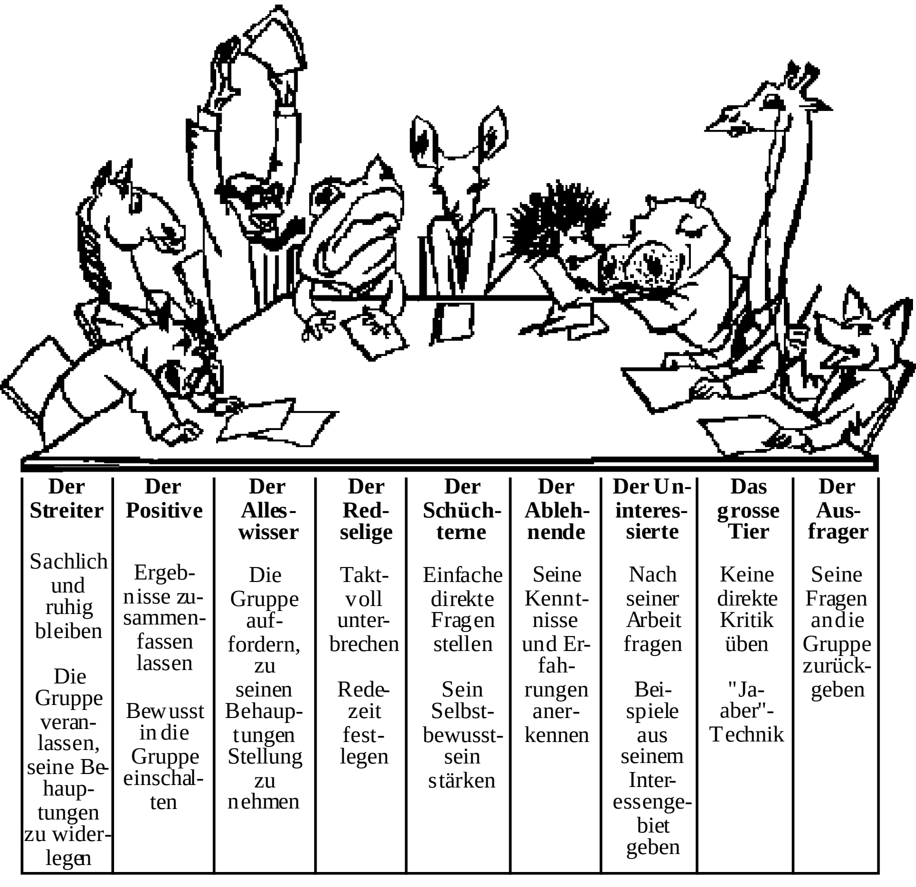
- Zusammenfassen, Visualisierung, Standort bestimmen;
- Diskussion, Visualisierung.

(6) Zusammenfassung, Visualisierung der Ergebnisse, evtl. Ergebnispräsentation von Arbeitsgruppen.

(7) Konsequenzen; Umsetzung; weiteres Vorgehen, Aufgabenverteilung.

Ein *gemeinsames* Mittagessen fördert Kontakte und erlaubt Gespräche in einem informellen Rahmen!

Der Diskussionsverlauf sollte vom Moderator optisch aufgezeichnet werden. Diese *Visualisierung* (z.B. mittels Flipchart, Metaplan nach Mauch 1981, Overhead, Pinwand usw.) dient als 'roter Faden' der Diskussion; Wiederholungen werden vermieden, Redundanzen aufgedeckt, kein Gedanke geht verloren, und Komplexität kann reduziert werden. Ausserdem entsteht dadurch gleichzeitig ein grafisches Protokoll. Die Dokumentation kann durch Protokollführung, Tonbandaufnahmen, Fotos und Videoaufnahmen ergänzt werden.



Der Streiter	Der Positive	Der Alleswisser	Der Redselige	Der Schüchterne	Der Ablehnende	Der Uninteressierte	Das grosse Tier	Der Ausfrager
Sachlich und ruhig bleiben Die Gruppe veranlassen, seine Behauptungen zu widerlegen	Ergebnisse zusammenfassen lassen Bewusst in die Gruppe einschalten	Die Gruppe auffordern, zu seinen Behauptungen Stellung zu nehmen	Taktvoll unterbrechen Redezeit festlegen	Einfache direkte Fragen stellen Sein Selbstbewusstsein stärken	Seine Kenntnisse und Erfahrungen anerkennen	Nach seiner Arbeit fragen Beispiele aus seinem Interessengebiet geben	Keine direkte Kritik üben "Ja-aber"-Technik	Seine Fragen an die Gruppe zurückgeben

Abbildung 14: Verhaltensstile sowie darauf abgestimmte Verhaltensweisen des Moderators.

4.5.3 Moderatorenschulung

Erfahrungen bisheriger Workshops haben die *grosse Bedeutung der Rolle des Workshopleiters bzw. Moderators und seiner Fähigkeiten, speziell seiner sozialen Fertigkeiten*, für das Gelingen des Workshops gezeigt. Softwareentwickler und Benutzer haben aber in der Regel keine Ausbildung hinsichtlich der Anleitung von Arbeitsgruppen bzw. der Leitung von Diskussionsgruppen erhalten und fühlen sich in derartigen Situationen vielfach auch unsicher. Ausserdem neigen Entwickler auch häufig zur Verteidigung ihrer Produkte und wehren Benutzervorschläge mit technischen Argu-

menten ab, was alles den positiven Ablauf eines Workshops und die Zielerreichung gefährden kann. Manchmal ist der Einsatz einer 'neutralen' Person (weder Benutzer noch Entwickler, evtl. Aussenstehender) als Moderator von Vorteil! Eine Moderatorenschulung sollte folgende Aspekte zumindest rudimentär behandeln. Die Moderatoren werden sich dann sicherer fühlen, und der Workshop wird problemloser und zufriedenstellender verlaufen!

• Verständnis der Moderatorenrolle:
  Ein Moderator ist weder ein Lehrer, noch ein Referent, noch ein Verkäufer eigener Ideen, sondern ein 'Vermittler', seine Hauptaufgabe besteht in der Strukturierung der Situation und des Ablaufs im Sinne der Zielerreichung!

• Gesprächsführung:
  Grundsätzlich *nichtdirektiv*; keine Verteidigung, keine Abwehr a priori (z.B. "geht nicht"); nicht eigene Hypothesen bestätigen oder durchsetzen wollen; aber: erklären (z.B. technische Probleme), realistische Erwartungen wecken.

• Diskussionsleitung:
  In der Diskussion anregen, steuern, ausgleichen, erläutern, fragen, rückfragen, hinterfragen, Bezüge herstellen, zusammenfassen, Details nicht ins Zentrum rücken, Konzentration auf das Wesentliche! Umgang mit verschiedenen 'Diskussionstypen' (siehe Abbildung 14).

4.5.4 Aufbereitung – Feedback – Umsetzung

Nach dem Workshop müssen die Ergebnisse *aufbereitet* und – zumindest in zusammengefasster Form – an die Teilnehmer *zurückgemeldet* werden. Ausserdem erreicht man eine enorme Motivationswirkung für die weitere Zusammenarbeit mit Benutzern, wenn z.B. kurzfristig mit geringem Aufwand realisierbare Vorschläge zur Systemverbesserung *umgesetzt* werden; die Benutzer erleben dadurch, dass ihre Beiträge ernst genommen werden und sie wirklich einen Einfluss auf die Systementwicklung haben!

4.6 BEFRAGUNG

4.6.1 Schriftliche und mündliche Befragung

Befragungen lassen sich wie folgt unterscheiden:

(1) Mündliche Befragung in Form des Interviews (siehe Richardson, Dohrenwend und Klein 1965; Stewart 1983, James 1989, Froschauer und Lueger 1992),

(2) schriftliche Befragung mit Hilfe von Fragebogen (siehe auch Mummendey 1987),

(3) schriftliche Befragung mit Hilfe eines 'interaktiven', elektronischen Fragebogens (z.B. per 'electronic-mail').

Folgende Aspekte sind bei einer Befragung zu berücksichtigen.

Strukturiertheitsgrad:

Je weniger man über ein Gebiet weiss, je weniger präzise Vorstellungen oder Hypothesen existieren, desto geringer strukturiert muss die Befragung sein – und umgekehrt. Eine grobe Unterscheidung kann getroffen werden zwischen

(1) nichtstandardisierter Befragung,

(2) teilstandardisierter Befragung und

(3) standardisierter Befragung.

Typen von Fragen:

In Abhängigkeit vom Strukturiertheitsgrad können die einzelnen Fragen im Interview bzw. im Fragebogen wie folgt unterschieden werden. Es gibt:

(1) *Offene Fragen*, das heisst, Fragen ohne vorgegebene Antwortmöglichkeiten,

(2) *geschlossene Fragen* mit unterschiedlichen Antworttypen (Alternativfragen [z.B. 'ja' / 'nein'], Mehrfachwahlfragen, Skalafragen usw.),

(3) *direkte Fragen* (Sinn und Zweck der Frage ist erkennbar),

(4) *indirekte Fragen* (Sinn und Zweck der Frage ist verschleiert, um bewusst verfälschte Antworten zu vermeiden).

Formulierungsregeln:

Der Erfolg einer Befragung hängt u.a. von der Qualität der Fragen ab. Nachfolgend sind einige Regeln aufgeführt, die beachtet werden sollten:

(1) Einfache Formulierungen wählen,

(2) lange Fragen vermeiden,

(3) Suggestivfragen vermeiden,

(4) nur eindeutig beantwortbare Fragen stellen (z.B. keine Doppelfragen),

(5) doppelte Verneinung vermeiden,

(6) Überforderung des Antworters durch zu umfangreiches Material vermeiden,

(7) konkrete Fragen sind besser als allgemeine.

Erhebungen in Form von *schriftlichen Befragungen* haben gegenüber persönlichen und telefonischen Interviews folgende Vorteile:

(1) Die befragte Person kann den Fragebogen dann ausfüllen, wenn sie Zeit dazu hat, allenfalls auch in mehreren Schritten zu verschiedenen Zeitpunkten.

(2) Die Situation bei der Beantwortung ist gegebenenfalls vollständig anonym und vertraulich.

(3) Erläuternde Unterlagen können, auch wenn sie komplex sind, ohne Zeitdruck durchgelesen werden.

(4) Nicht zuletzt entfällt der Kontaktierungs- und Befragungsaufwand beim Erheber, was zu einem kostengünstigen Vorgehen führt.

Wichtig bleibt festzuhalten, dass bei betrieblichen Befragungen die Befragten *während* der Arbeitszeit ausreichend Gelegenheit zum Ausfüllen der Fragebogen erhalten.

Zu den wesentlichen Nachteilen der schriftlichen Befragung gehört, dass über Gründe und Motive derjenigen, die nicht antworten, keinerlei Informationen vorliegen. Erschwerend kommt bei schriftlichen Befragungen hinzu, dass die Rücklaufquoten – je nach Zielpublikum, Gültigkeit und Anzahl von Adressen, Streuung des Fragebogens und Administration des Mahnwesens – stark unterschiedlich sein können; im allgemeinen sind Rücklaufquoten unter 30% als kritisch, 30–50% als akzeptabel und über 50% als gut bis sehr gut anzusehen. Niedrige Rücklaufquoten können dann vorkommen, wenn der Fragebogen wenig ansprechend gestaltet ist, eine grosse Zahl von Fragebogen an ein breitgefächertes Zielpublikum verschickt wird (z.B. bei Umfragen in Zeitschriften) oder die Adressen nicht mehr auf dem aktuellen Stand sind. Dagegen können bei regelmässigen telefonischen Befragungen über 60% des Zielpublikums erreicht und inter-

viewt werden. Eine ausführliche Einführung in das Gebiet der schriftlichen Befragung geben Mummendey (1987) und Vögele (1992).

Bei einer *geringen Rücklaufquote* sind die Ursachen hierfür ausfindig zu machen. Der beste und in der Praxis auch am häufigsten beschrittene Weg, um hier mehr Informationen erhalten zu können, besteht in einer direkten telefonischen Nachfrage bei einer repräsentativen (z.B. zufällig ausgewählten) Stichprobe aller Nichtantwortenden. Da nur das Motiv der Nichtteilnahme sowie einige grundlegende zusätzliche Angaben interessieren, kann das Telefongespräch kurzgehalten werden; die Kosten für eine derartige Nacherhebung sind daher eher gering.

Für eine sauber durchgeführte Nacherhebung bedarf es jedoch einer gut organisierten Rücklaufadministration, bei der eindeutig festgestellt werden kann, wer nicht – bzw. noch nicht – geantwortet hat. Damit ist keineswegs die Vertraulichkeit der bei der postalischen Befragung gemachten Angaben in Frage gestellt. Entscheidend ist dabei, dass der Auftraggeber der Umfrage keine Rückschlüsse auf einzelne Antwortende ziehen kann. Bei einer solchen Nachbefragung von 300 Nichtteilnehmern aus einer Umfrage unter 5000 Personen ergaben sich die folgenden Begründungen für eine Nichtteilnahme: 40% hatten 'keine Zeit', 30% gaben an, 'den Fragebogen nie gesehen zu haben', 10% fühlten sich 'nicht zuständig', für 5% war 'der Fragebogen zu umfangreich', 20% hatten andere Begründungen.

4.6.2 Befragung in elektronischer Form

Befragungen auf elektronischem Wege (email) sind – abgesehen von der Programmerstellung des 'interaktiven' Fragebogens – gegenüber schriftlichen Befragungen nahezu kostenlos. Dafür können jedoch grundsätzlich nur Personen erreicht werden, welche über einen entsprechenden elektronischen Anschluss verfügen.

Beim Einsatz von 'interaktiven' Fragebogen in elektronischer Form sind folgende positiven und negativen Aspekte zu berücksichtigen:

- Es gibt keine Kontrolle darüber, ob eine Person mehrmals antwortet.
- Nur Benutzer mit Zugang zu einem Computer mit email-Anschluss können befragt werden.
- Es sind im Netzwerk und im 'interaktiven' Fragebogen besondere softwaretechnische Sicherheitsvorkehrungen gegen Eindringlinge von 'aussen' zu treffen.

– Da ein elektronischer Fragebogen kein normaler Bogen aus Papier ist, der geduldig auf dem Schreibtisch auf seine Beantwortung wartet, bedarf es zusätzlicher Motivatoren, um die Bearbeitung in den Aufmerksamkeitsfokus des Benutzers zu rücken.

+/- Es bedarf einer guten Informationspolitik im organisatorischen Vorfeld.

+/- Frageschleifen mit stets ähnlichen Fragen sollten vermieden werden ('keep it simple').

\+ Programmierbare Verzweigungen lassen den Fragenverlauf in Abhängigkeit von den konkreten Antworten von überflüssigen Fragen freihalten.

\+ Die Abspeicherung der Antworten kann gleich in einem für ein Statistikauswertungsprogramm passenden Format erfolgen.

\+ Bei regelmässigem Einsatz sind die Kosten für die Datenerhebung und -auswertung verschwindend gering.

\+ Es kann ein beliebig grosser Benutzerkreis angesprochen werden.

4.7 DAS PFLICHTENHEFT

Das Pflichtenheft umfasst in der Regel alle Anforderungen an das zu erstellende System (Hard- und Software); die Anforderungen umfassen die funktional-technischen, die hard- und softwareergonomischen sowie die wirtschaftlichen Aspekte (siehe Schreiber 1994). Das Pflichtenheft ist in der Regel als Textdokument gegeben. Um jedoch unter allen Betroffenen ein möglichst gemeinsames Verständnis zu ermöglichen, empfiehlt es sich, die Inhalte des Pflichtenheftes zusätzlich auch in anschaulicher Form darzustellen. Bevor man jedoch mit formalen Beschreibungssprachen arbeitet, sind einfachere und anschaulichere Verfahren angesagt: z.B. Simulationsmethoden und Prototyping.

4.8 SIMULATIONSMETHODEN

Unter Simulationsmethoden werden hier alle Verfahren zusammengefasst, welche Attrappen verschiedenster Art verwenden und für ihre Erstellung und Anwendung möglichst wenig Aufwand erfordern (je einfacher und preiswerter, desto besser! Siehe auch Ehn und Kyng 1991). Um diesem Anspruch in den jeweiligen Arbeitssitzungen, Workshops, Benutzertreffen usw. nachkommen zu können, wird von allen Beteiligten ein ausreichendes Mass an Offenheit, Neugierde und Kreativität benötigt.

Kasten 19: Ein Beispiel für eine 'Wizard of oz'-Simluationsstudie.

In einem grossen europäischen Forschungsprojekt aus dem Jahre 1990 zur Entwicklung eines Datenbankabfragesystems mit einer natürlichsprachlichen Ein- und Ausgabe wurde folgende Simulationsstudie ('wizard of oz') durchgeführt: Ein normales Bildschirmgerät im Lesesaal einer Bibliothek, welches ursprünglich für die Literaturrecherche mittels einer traditionellen Bibliothekssoftware mit Menüoberfläche eingesetzt wurde, bekam eine 'natürlichsprachliche' Benutzungsoberfläche; dies wurde den Benutzern auf entsprechenden Hinweistafeln mitgeteilt. In der Tat wurde das Bildschirmgerät jedoch lediglich mit einem anderen Bildschirmgerät im Raum eines Bibliotheksangestellten nebenan verbunden, welcher dann die Eingaben der Benutzer las und über ein zweites, traditionelles Gerät die entsprechenden Recherchen durchführte. Die Ergebnisse wurden dann vom Angestellten 'von Hand' auf dem simulierten Bildschirmgerät ausgegeben.

Die Ergebnisse waren verblüffend: Die natürlichsprachlichen Eingaben der Benutzer waren oft grammatikalisch falsch, mit einer schlechten Satzstellung und weitgehend dürftig formuliert; es traten sehr viele Buchstabierfehler auf, dagegen erstaunlich wenig Fragesätze; ein hoher Prozentsatz der Sätze enthielt kein Verb oder bestand sogar nur aus komplexen Hauptwortkonstruktionen; die Benutzung von Fürwörtern war auf einige wenige beschränkt. Einige Benutzer fingen sogar an, dem System Fragen zu politischen Problemen des Weltgeschehens (z.B. "Was macht Gorbatschow gerade?") – bzw. anderen nicht bibliotheksbezogenen Problemen – zu stellen.

Es muss nicht immer die Hochglanzfolie sein, wenn es schon eine einfache Handskizze tut. Am besten bewähren sich Medien (z.B. [Wand-]Tafeln), auf denen man gemeinsam zeichnen und beliebig verbessern kann. Für die Gestaltung grafischer Oberflächen gibt es bereits spezielle Methoden (z.B. PICTIVE von Muller 1993). Das wichtigste ist jedoch, dass die Beteiligten anfangen, gemeinsam etwas zu analysieren, zu bewerten und zu verbessern.

Im Kasten 19 ist eine Simulationsstudie beschrieben, welche durchgeführt wurde, um das zu erwartende Benutzerverhalten bei der geplanten System-

version im vorhinein studieren zu können. Es zeigten sich nicht vorhergesagte Erwartungshaltungen seitens der Benutzer.

4.8.1 Szenarium

Szenarien sind Situationsbeschreibungen über relevante, gegenwärtige oder angestrebte Arbeitssituationen und umfassen: auszuführende Handlungen, eingesetzte oder einzusetzende Arbeitsmittel und die Motivation bzw. Gründe für die einzelnen Handlungen, Strukturen usw. Ein Szenario ist durch die folgenden Merkmale gekennzeichnet:

(1) Es geht um *einen einzelnen* realen – oder fiktiven – Benutzer.

(2) Von diesem Benutzer wird im Rahmen seiner Aufgabenbearbeitung eine bestimmte Anzahl an modernen Werkzeugen eingesetzt.

(3) Es soll bei der beschriebenen Aufgabenbearbeitung ein ganz bestimmtes Arbeitsergebnis erzielt werden.

(4) Dies alles spielt sich in einer konkret beschriebenen Arbeitsumgebung unter genau angegebenen Bedingungen ab.

Szenarien werden in der Regel von ein bis zwei Personen erstellt, welche ganztägig für zirka zwei Wochen daran arbeiten. Dazu gehören: Auswahl eines Problem- bzw. Arbeitsbereiches; Befragung der Fachanwender oder anderer relevanter Ansprechpartner; Brainstorming über die Verwendung und den möglichen Einsatz neuer Technologien; Beschreiben des Szenarios auf ein bis zwei Seiten in einem erzählenden Stil, welcher möglichst lebendig, realistisch und gut nachvollziehbar wirkt; technische und andere Details können im Anhang zum Szenario dargestellt werden. Jeder, der das Szenarium gelesen hat, sollte anschliessend mit eigenen Worten erklären können, worum es dabei geht und was die wesentlichen neuen Ideen und Aspekte sind (siehe auch bei Meyer-Schönherr 1992).

Da bei einem einzelnen Szenario die Gefahr besteht, dass es zu stark auf bestimmte Vorstellungen fokussiert und damit in seiner Aussagekraft eingeschränkt ist, empfiehlt es sich, mehrere Szenarien anzufertigen. Entwickler haben dann durch solche Szenarien die Möglichkeit, sich einen detaillierten Einblick in zukünftige Anwendungen der Benutzer zu verschaffen.

4.8.2 Drehbuch

Während Szenarien sich eher auf das gesamte Arbeitsumfeld beziehen, lassen sich *Drehbücher* ('storyboards'; siehe Andriole 1989) als Simulations-

methoden für einzelne Abläufe einsetzen. Nicht wie im Film – obwohl der Begriff von dort übernommen wurde – sind Drehbücher ein Textdokument, sondern vielmehr eine Abfolge von Bildern (z.B. 'mock-ups', Handskizzen usw.). Hat man z.B. einen Grafikeditor auf einem PC zur Hand und zusätzlich noch ein Datenbankprogramm, welches Bilder verwalten kann, so lässt sich mit verhältnismässig geringem Aufwand ein ganzer Komplex von Abläufen erzeugen und nach verschiedenen Kriterien zusammenstellen. Selbst wenn man diese Werkzeuge nicht hat, so kann man mit ganz einfachen Papier- und Bleistiftskizzen arbeiten, welche kopiert, verändert, überarbeitet und durchnumeriert werden. Werden Szenarien mit Drehbüchern gekoppelt, so kann daraus eine *Videodemonstration* entwickelt werden. Bei innovativen Problemstellungen, für die es noch keine vergleichbaren Lösungen gibt, ist eine echte *Simulationsstudie* oftmals die einzige Möglichkeit, um das Benutzerverhalten im vorhinein studieren zu können (siehe Kasten 19).

4.9 FORMALE DARSTELLUNGSMETHODEN

Bei informations- und aktivitätsorientierten Darstellungsmethoden aus der Informatik wird in formalisierter Form die Veranschaulichung des geplanten Systems verfolgt. Dafür werden gleichermassen Entscheidungstabellen, Konfigurations-, Funktions-, Aktivitäten-, Datenflussablauf- und Programmablaufdiagramme wie z.B. HIPO, JACKSON, YOURDAN, Petri-Netze, SADT usw. eingesetzt (speziell für interaktive Systeme siehe Dix 1991). Allgemein zeichnen sich die Analyse- und Darstellungsmethoden aus der Informatik durch eine zum Teil sehr komplexe Syntax und abstrakte Aktivitäteneinteilung aus und sind in diesem Sinne meist nicht als 'beteiligungsorientiert' zu bewerten. Sie eignen sich am ehesten für die Zusammenarbeit von EDV-Fachleuten (siehe Martin und McClure 1985 sowie Raasch 1991) und sind für eine saubere Spezifikation oft unentbehrlich.

Der Modellansatz von Oberquelle (1987) liefert nun einen Beitrag zur Entwicklung einer *benutzerorientierten Beschreibungssprache* für Organisationen mit interaktiven Computeranwendungen. Dieser Ansatz ermöglicht die Darstellung der abstrakten Gesamtarchitektur einschliesslich der Benutzungsmodelle von computergestützten Teilen aus der Sicht der Benutzer, um den Softwareentwicklungsprozess in seiner Gesamtheit unterstützen zu können. Es werden unter anderem Rollen-Funktions-Aktions-Netze (RFA-Netze) eingeführt, welche Arbeitsorganisationen mit interaktivem Rechnereinsatz beschreiben können.

4.10 PROTOTYPING

Während in der Produktgüterindustrie ein Prototyp – von Designstudien abgesehen – in der Regel ein voll funktionstüchtiges und fehlerfreies Produkt darstellt, ist ein Prototyp im Bereich Software ein irgendwie unfertiges Produkt: Sei es, dass die eigentliche Funktionalität oder die eigentliche Oberfläche fehlt, sei es, dass das Antwortzeitverhalten unzureichend ist, oder sei es, dass die interne Programmstruktur nicht nach gängigen softwaretechnischen Kriterien ausgestaltet ist (siehe Pomberger und Blaschek 1993). Jedenfalls sollte ein Softwareprototyp *operational* sein; ob diese Operationalität direkt gegeben sein muss oder auch simuliert werden kann, wird unterschiedlich beurteilt. Softwareprototyen sollten im Rahmen einer Prototypingstrategie *schnell* und *kostengünstig realisiert* werden können.

Prototyping ist im Kontext partizipativer Softwareentwicklung eine Spezifikationsmethode, bei der die softwareergonomischen Anforderungen zur Benutzungsoberfläche präzise festgelegt werden können (siehe Floyd 1984). Darüber hinaus kann ein Prototyp zur Erprobung und Evaluation von Lösungsvorschlägen, zur Beurteilung der Qualität des Entwurfes einer Systemarchitektur und zur Prüfung verschiedener Realisierungsmöglichkeiten von Systemkomponenten eingesetzt werden. Im Unterschied zu einem (nur simulierten) Softwaremodell ist ein Prototyp immer ein Stück betriebsfähige Software.

4.10.1 Horizontales und vertikales Prototyping

Während ein *horizontaler Prototyp* eine weitgehend funktionslose Fassade ('mock up') darstellt, umfasst ein *vertikaler Prototyp* nur das rein funktionale Modell ohne die angestrebte Benutzungsoberfläche (siehe Abbildung 15). Horizontale Prototypen eignen sich hervorragend für Benutzer-Entwickler-Diskussionen. Vertikale Prototypen dagegen eignen sich eher zum Austesten von Anwendungsfunktionen im Rahmen von Machbarkeitsstudien. Wenn ein zunächst unvollständiger Prototyp zum vollständigen Zielsystem ausgebaut wird, spricht man von einem Pilotsystem *(evolutionäre Prototypingstrategie).* Mit immer mächtiger werdenden Entwicklungsumgebungen, insbesondere bei objektorientierter Programmierung – ggf. zusammen mit einer objektorientierten Spezifikationsmethode –, kann Prototyping fliessend in die eigentliche Systementwicklung übergehen.

Prototyping als eine benutzernahe Methode für den Softwareentwicklungsprozess hat inzwischen eine breite Verwendung gefunden (Kieback et al. 1991) und wird von vielen Unternehmen schon regelmässig eingesetzt.

Prototyping wird dabei hauptsächlich zur methodischen Unterstützung von Konzeptions- und Spezifikationsaufgaben (Quadrant-II) herangezogen und dient vorwiegend – beim horizontalen Prototyping – der Abklärung von Fragen zur Benutzungsoberfläche. Insgesamt wird das Vorgehen mit Prototyping von den Unternehmen als *sehr nützlich* bewertet, da es eine sinnvolle Problemlösemethode für den Softwareentwicklungsprozess darstellt und dabei vor allem auch die Zusammenarbeit mit den Benutzern erleichtern kann. Mit dem Einsatz von Prototyping gehen signifikant geringere Kosten- und Terminüberschreitungen einher! (vgl. Rauterberg und Strohm 1992).

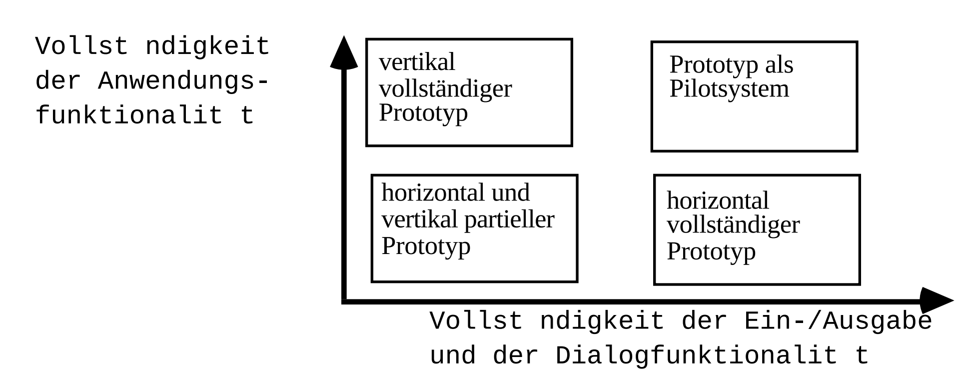


Abbildung 15: Übersicht über die verschiedenen Arten von Softwareprototypen.

Eine Voraussetzung für den Einsatz des Prototyping bilden Softwarewerkzeuge, die eine schnelle und ökonomische Produktion von Prototypen erlauben. In diesem Zusammenhang sind Masken- bzw. Reportgeneratoren, Dialogsystem-, Programm- und Datengeneratoren als wichtige Voraussetzungen für die effiziente Gestaltung eines iterativ-zyklischen Entwicklungsprozesses zu nennen. Endbenutzerorientierte Werkzeuge, die die Produktion von Systembeispielen durch den Benutzer selbst ermöglichen, gewinnen immer stärker an Bedeutung. Im Rahmen von Sprachen der vierten Generation werden vermehrt integrierte Entwicklungssysteme vorgestellt, die gleichermassen das Projektmanagement, die Analysen, den Entwurf und die Systemrealisierung unterstützen sollen.

Erfahrungen bei der Verwendung von Prototypen zeigen, dass bei der Evaluation von Entwurfsvarianten zusammen mit Benutzern eine Menge an

neuartigen Designvorschlägen für den Softwareentwickler anfällt, allerdings auf eine eher unsystematische Weise und durchaus nicht immer nur auf die eigentlich diskutierten Probleme bezogen. Viele dieser Hinweise beziehen sich auf Aspekte, die mit dem Prototypingwerkzeug z.T. gar nicht modelliert werden können. Darunter fallen u.a. Spezifikationen der Funktionalität einzelner Systembefehle. Diese Hinweise müssen dann extra schriftlich festgehalten werden und können erst später (z.B. bei der Programmierung des Systems) berücksichtigt werden. Ein wesentlicher Vorteil des Prototyping besteht darin, dass die Kommunikation zwischen Benutzern, Analytikern und Entwicklern auf eine für alle Beteiligte anschauliche Art angeregt und aufrechterhalten wird.

4.10.2 Probleme beim Prototyping

Bei der Evaluation von Prototypen unter Einbezug der Benutzer stellt sich die Frage nach der Gewichtung und Interpretation der Ergebnisse. Dabei treten verschiedene Probleme auf:

(1) Die Ergebnisse sind kaum quantifizierbar; es lässt sich nicht immer feststellen, ob eine Entwurfsvariante besser ist als eine andere.

(2) Die Aussagen verschiedener Benutzer können widersprüchlich sein – insbesondere über mehrere Evaluationszyklen hinweg.

(3) Es kann durchaus vorkommen, dass die Benutzer bei der Evaluation von Entwürfen in bezug auf die Aufgabengestaltung gegen ihre eigenen Interessen entscheiden, indem sie ihre eigenen Aufgaben möglichst vereinfachen und ihre eigenen Handlungsspielräume einschränken. Sie handeln sich damit für ihre spätere Arbeit Nachteile ein.

Das *erste Problem* tritt beim Bewerten mehrerer alternativer Entwürfe auf, wenn eine beste Variante bestimmt werden soll. Ziel des Entwurfs und der Evaluation zusammen mit den Benutzern ist aber nicht primär die endgültige Bewertung von Entwürfen, sondern das Erarbeiten einer *guten* Lösung. Jeder Entwurf hat seine Stärken und Schwächen, eine brauchbare Lösung lässt sich häufig durch eine Kombination einzelner Elemente der verschiedenen Entwürfe zusammenstellen.

Ein *zweites Problem* sind die widersprüchlichen Aussagen und Veränderungsvorschläge der Benutzer. Sie sind häufig ein Indikator dafür, dass der zu beurteilende Entwurf von einer zu formalisierten Auffassung der Arbeitsaufgabe ausgeht und die Freiheitsgrade der 'widersprüchlichen' Benutzer unnötig einschränkt. Es sollte in diesem Fall nach einer Lösung gesucht

werden, welche durch eine weitergehende Differenzierung genügend flexibel und individuell anpassbar ist, um die möglicherweise widersprüchlichen Vorschläge verschiedener Benutzer zu vereinen.

Der *dritte Problembereich* dagegen ist ein methodisches Problem, welches bei der Evaluation von Prototypen ohne die Beteiligung von Arbeitsgestaltern auftreten kann. Da die in der Regel nicht funktionstüchtigen Prototypen von den Benutzern nur über einen kurzen Zeitraum hinweg evaluiert werden, stehen primär Fragen der Erlernbarkeit im Vordergrund. Die Benutzer – insbesondere Computerneulinge – werden Entwürfe vorziehen, welche eine starke Führung durch das System aufweisen. Solche Systeme erweisen sich aber nach einigem Gebrauch (Tage bis Wochen) als zu inflexibel und werden dann von den Benutzern – insbesondere den an der Entwicklung nicht beteiligten – abgelehnt. Hier kann der Einsatz von objektiven arbeitspsychologischen Analyseverfahren durch professionelle Arbeitsgestalter Abhilfe schaffen.

Bei der Auswahl von Prototypingwerkzeugen ('interface-development-tools') kann die Checkliste von Hix und Schulman (1991) behilflich sein.

4.10.3 Stärken und Schwächen von Prototyping

Nachweisbare Stärken des Prototypingansatzes sind

+ Reduktion der Kostenüberschreitungen,
+ Reduktion der Terminüberschreitungen,
+ schnellere Verfügbarkeit der Software,
+ besseres Verständnis der Benutzer für die Möglichkeiten und Grenzen der Anwendung,
+ benutzungsgerechte Lösungen, insbesondere für die Dialog- und Anwendungsschnittstelle.

Bekannte Schwächen, welche sich jedoch bei adäquatem Vorgehen vermeiden lassen, sind

– Quick-and-dirty-Entwicklungen statt sauber programmierte Lösungen,
– zu langes Festhalten am Prototyp und dem einmal eingeschlagenen Weg,
– bei komplexen Applikationen ist kein effektives Prototyping möglich,

– Qualifizierungserfordernisse für die Analytiker bzw. Programmierer hinsichtlich der Prototypingwerkzeuge,

– unzureichende Prototypingwerkzeuge (siehe Übersicht 2 im Anhang).

4.11 CHECKLISTEN

Checklisten sind speziell für die Bewertung der Benutzungsfreundlichkeit entwickelt worden und zumeist nur in englischen Fassungen erhältlich. Deutschsprachige Checklisten sind in Spinas et al. (1983), Baitsch et al. (1989), Duell und Katz (1990), Koch, Reiterer und Gärtner (1990) sowie Siemens Nixdorf Styleguide (1992) zu finden (siehe auch Übersicht 2 im Anhang).

Stärken:

+ Gegenüber Benutzungstests verringert sich der Bewertungsaufwand beim Einsatz von Checklisten beträchtlich, womit das wirtschaftliche Interesse an diesen Verfahren gegeben ist.

+ Checklisten helfen zu verhindern, dass wichtige Aspekte bei der Bewertung vergessen werden.

+ Checklisten eignen sich besonders als Hilfsmittel bei der Bearbeitung von Routinefällen.

+ Checklisten enthalten in komprimierter Form die als relevant anzusehenden Aspekte.

Schwächen:

– Checklisten können Sachverstand nicht ersetzen; das Arbeiten mit ihnen bedingt hinreichende Kenntnis von Aufgaben, Problemen, Zusammenhängen und Lösungsansätzen.

– Die Zuverlässigkeit ist geringer als bei anderen Verfahren, da bewusstes oder unbewusstes Missverstehen der Fragen möglich ist.

– Nur begrenzte Möglichkeiten zur eingehenderen Erläuterung der einzelnen Fragen sind bei Papierversionen gegeben.

– Spezielle Aspekte, die nicht in das Raster der Checkliste hineinpassen, lassen sich nicht adäquat bei der Auswertung berücksichtigen.

4.12 EVALUATIONSTECHNIKEN

Bewertungs- und Evaluationstechniken sind im vorliegenden Zusammenhang formalisierte Verfahren, die die zu beurteilenden Systeme hinsichtlich ihrer Eigenschaften in eine Rangordnung bez. der angestrebten Benutzungseigenschaften bringen. Diese Rangordnung kann dann als Ausgangsbasis für eine Entscheidung dienen. Die Formalisierung der Bewertungsverfahren strebt folgende Ziele an:

(1) Die Vergleichbarkeit der Ergebnisse über alle bewerteten Systeme hinweg,

(2) die Nachvollziehbarkeit des Bewertungsvorganges.

Die verschiedenen Bewertungstechniken lassen sich hinsichtlich

(1) ihres Aufwandes,

(2) ihres Formalisierungsgrades und

(3) der Messbarkeit der verwendeten Bewertungsdimensionen unterscheiden.

Für eine Übersicht über die verschiedenen Reviewverfahren und ihre Handhabung sei auf Freedman und Weinberg (1990) sowie Frühauf, Ludewig und Sandmayr (1991b) verwiesen. Die Beteiligung von Endbenutzern bei Reviewsitzungen wird von Freedman und Weinberg (1990) explizit gefordert. Die Art der Beteiligung hängt vom Review und der fachlichen Qualifikation der Beteiligten ab.

Mit dem EVADIS-Verfahren (Oppermann et al. 1992) steht ein Instrument zur Verfügung, mit dem die Qualität einer Benutzungsoberfläche bewertet werden kann. Es handelt sich dabei um einen Fragenkatalog, der verschiedene Prüfkriterien enthält. Fragen werden nach einem festgelegten Ablauf beantwortet und ausgewertet. EVADIS-II wurde mit dem Ziel entwickelt, Softwaresysteme für den Büro- und Verwaltungsbereich zu evaluieren. Die Benutzer sind vornehmlich "Arbeitende in diesem Bereich, wie beispielsweise Schreibkräfte, Sachbearbeiter oder Wissensverarbeiter (Fach- oder Führungskräfte)". Es kann auf Softwareprodukte wie Datenbanken, Tabellenkalkulation, Bürografik, Textverarbeitung und elektronische Post angewendet werden.

Das EVADIS-II-Verfahren versteht sich als "Versuch, die relevanten ergonomischen Eigenschaften von Dialogsystemen für den Büro- und Verwaltungsbereich in einer ganzheitlichen Weise zu erfassen". Ganzheitlich me-

int, dass "möglichst alle benutzer- und aufgabenrelevanten Eigenschaften der Schnittstelle in allen Teilen des fraglichen Arbeitssystems" ermittelt werden sollen. "Es geht also nicht nur um die Erwartungskonformität einer Schnittstelle, nicht nur um ihre Steuerbarkeit, um zwei allgemeine Kriterien für Schnittstellengüte zu nennen. Es geht auch nicht allein um die Leistungsfähigkeit und Güte einzelner Systemkomponenten, wie z.B. eines Hilfesystems. Ganzheitlichkeit bedeutet auf der anderen Seite allerdings auch nicht, dass das Verfahren beanspruchen könnte, Schnittstellengüte abschliessend und erschöpfend darzustellen." (Oppermann et al. 1992)

Für EVADIS-II sind hauptsächlich die folgenden drei Anwendungsmöglichkeiten vorgesehen:

(1) *Bewerten während der Systemgestaltung:*
Einbeziehung von EVADIS in den Entwicklungsprozess eines entstehenden Softwareproduktes für ein Bürosystem. Beispielsweise beim Einsatz von Prototyping. Mögliche Anwender sind Softwarehäuser oder EDV-Abteilungen.

(2) *Bewerten von Systemausprägungen:*
Bewertung eines fertigen Softwareproduktes für ein Bürosystem hinsichtlich der ergonomischen Qualität einzelner Systemausprägungen, beispielsweise im Rahmen eines Abnahmetests. Mögliche Anwender sind Auftraggeber von Software oder Softwareprüfstellen.

(3) *Bewerten zum Zwecke eines Systemvergleichs oder einer Marktorientierung:*
Bewertung mehrerer fertiger Softwareprodukte für ein Bürosystem im Rahmen eines Systemvergleiches. Beispielsweise vor einer Kaufentscheidung von Standardsoftware. Mögliche Anwender sind Auftraggeber von Software, Softwareprüfstellen, Konsumentenberatungseinrichtungen oder Technologieberatungsstellen von Gewerkschaften.

Der Einsatz des EVADIS-Verfahrens setzt hinreichende Systemkenntnisse der zu evaluierenden Software voraus.

4.13 CODEINSPEKTION

Da in der Realisierungsphase das zu erstellende System ausprogrammiert wird, ist die Beteiligung der Endbenutzer auf ein Minimum reduziert. Als Beteiligungsmethoden kommen Reviewsitzungen mit Codeinspektion (Freedman und Weinberg 1990) in Frage. Dabei können die Benutzer – bei entsprechender fachlicher Qualifikation – neben der Beurteilung der Ver-

ständlichkeit und Korrektheit des erzeugten Source- und Objektcodes auch für die Zusammenstellung der benötigten Testdatensätze mitverantwortlich sein.

Obwohl – im Idealfall – ein ausreichender Aufwand in den frühen Phasen investiert worden ist, kann es dennoch vorkommen, dass offene Fragen und Probleme erst bei der Realisierung auftreten. Hierbei hat es sich als besonders nützlich erwiesen, wenn möglichst grosse Teile der Entwicklung am Ort der Arbeitsstätte durchgeführt werden können, um somit möglichst schnelle Rückfragen zwischen Benutzern und Entwicklern zuzulassen. Die Ergebnisse dieser Rückfragen sollten festgehalten und in den nächsten Reviewsitzungen angesprochen werden.

4.14 Benutzungs- und Usability-Tests

Benutzungs- bzw. Usabilitytests sind eine Methode, um speziell die *interaktiven Eigenschaften* von Softwareprodukten zu erfassen. Sie lassen sich in zwei Klassen unterscheiden: *aufgabenorientierte* und *benutzungsorientierte* Tests (siehe auch Dumas und Redish 1993 sowie Nielsen 1993). Die rein technischen Tests werden nicht weiter besprochen (siehe hierzu Frühauf et al. 1991b). Die aufgabenorientierten Benutzungstests sind bei der Evaluation eines (z.B. vertikalen) Prototyps zur Gewinnung von Gestaltungs- und Verbesserungsvorschlägen bzw. zur Analyse von Schwachstellen in der Benutzbarkeit einsetzbar. Jeder Test wird durch einen oder zwei Testleiter vorbereitet und durchgeführt. Es empfiehlt sich, dass ein Produktverantwortlicher und ein Repräsentant der Entwicklungsabteilung als passive Beobachter beteiligt sind, um die Vermittelbarkeit der Ergebnisse zu gewährleisten.

Benutzungstests werden in der Regel in einem speziell eingerichteten Labor durchgeführt (siehe Nielsen 1994). Sie können aber auch direkt am Arbeitsplatz erfolgen. Benutzungsorientierte Tests lassen sich in zwei Gruppen unterteilen:

Induktive ('formative evaluation') und *deduktive* ('summative evaluation') Benutzungstests.

Die induktiven Benutzungstests sind bei der Evaluation einer (Vor-)Version zur Gewinnung von Gestaltungs- und Verbesserungsvorschlägen bzw. zur Analyse von Schwachstellen in der Benutzbarkeit einsetzbar. Induktive Benutzungstests können immer dann zum Einsatz kommen, wenn nur *eine* Version der zu testenden Software vorliegt. Demgegenüber verfolgen de-

duktive Benutzungstests primär den Zweck, zwischen mehreren Alternativen (mindestens zwei Prototypen bzw. Versionen) zu entscheiden. Zusätzlich lassen sich jedoch auch mit deduktiven Benutzungstests Gestaltungs- und Verbesserungsvorschläge gewinnen.

4.14.1 Aufgabenorientierter Benutzungstest

Wie bei den Szenarien wird bei aufgabenorientierten Benutzungstests eine möglichst realistische Arbeitssituation nachgestellt. Im Unterschied zu den Szenarien handelt es sich nicht um eine verbale Beschreibung, sondern um das möglichst realistische Nachbilden der jeweiligen Arbeitssituation (siehe auch die 'use cases' bei Booch 1994). Die aufgabenorientierten Benutzungstests lassen sich z.B. in Benutzer-Entwickler-Workshops einsetzen, um eine typische Arbeitssituation im Rollenspiel nachzuspielen. Dadurch kann dem Entwickler eine direkte, anschauliche Rückmeldung über die zunächst verbal beschriebene Situation gegeben werden. Ein Benutzer übernimmt dabei seine Rolle als Mitarbeiter in einer typischen bzw. geplanten Arbeitssituation mit entsprechend abgesprochenen Aufgaben.

Unter Einsatz des erstellten Prototyps wird die Aufgabenbearbeitung durchgespielt. Wenn noch kein Prototyp vorhanden ist, kann dieser auch mit möglichst einfachen Hilfsmitteln simuliert werden (Pappschachtel als Computerattrappe, Folien oder Papierblätter als Maskenersatz usw.; siehe Ehn und Kyng 1991). Die anderen Benutzer übernehmen neben dem Testleiter die Rolle der Beobachter und Kommentatoren, um z.B. Unterschiede zu der von ihnen bevorzugten Bearbeitungsweise festzustellen. Am besten lässt sich ein aufgabenorientierter Benutzungstest direkt am Arbeitsplatz oder in unmittelbarer Nähe dazu durchführen.

4.14.2 Induktiver Benutzungstest

Bei der Durchführung eines induktiven Benutzungstests ist eine Anzahl von Bedingungen zu beachten. Da die meisten Bedingungen zur Durchführung eines induktiven Benutzungstests mit denen eines deduktiven Benutzungstests übereinstimmen, werden bei der Darstellung deduktiver Benutzungstests nur noch die Änderungen und Ergänzungen erwähnt. Um Artefakte bei der Benutzung – hervorgerufen durch unvollständige Systemfunktionalität – zu vermeiden, sollte im Bereich der zu testenden Systemfunktionen ein möglichst realitätsgerechtes Systemverhalten gegeben sein. Um die einzelnen Aktionen der Benutzer später auswerten zu können, empfiehlt es

sich, eine automatische Aufzeichnung der Tastendrucke in das Testsystem einzubauen.

Kasten 20: Beispiele für problemorientierte und handlungsorientierte Beschreibungen von Testaufgaben.

Problemorientierte Aufgabenbeschreibung:
"Erstellen Sie bitte einen Brief mit folgendem Inhalt ... und bereiten Sie ihn zum Eintüten und Versenden an folgende Adressen ... vor. Benutzen Sie dabei bitte als Briefvorlage das Dokument mit dem Namen ..."

Handlungsorientierte Aufgabenbeschreibung:
"Laden Sie bitte das Textdokument mit folgendem Namen ... und ergänzen Sie es um den folgenden Inhalt ... Versehen Sie dieses Textdokument mit allen für die Serienbrieferstellung notwendigen Steueranweisungen ... usw."

Als erstes sind eine oder mehrere Testaufgaben abgestimmt auf die zu testenden Systemteile festzulegen. Diese Testaufgaben sind dem typischen Aufgabenkontext des zukünftigen Endbenutzers zu entnehmen. Sofern dieser Aufgabenkontext nicht oder nur vage bekannt ist, sollten die Testaufgaben zumindest jedoch typische Teilaufgaben enthalten. Eine einzelne Testaufgabe sollte nicht zu komplex, aber auch nicht zu einfach sein; die Bearbeitungszeiten sollten ungefähr zwischen fünf und dreissig Minuten liegen. Die Formulierung der Testaufgaben sollte den Benutzern während der Aufgabenbearbeitung als schriftliche Fassung zugänglich sein. Die Aufgabenbeschreibung hat das unterschiedliche fachliche Vorwissen der Benutzer zu berücksichtigen; bei hohem fachspezifischen Vorwissen ist die Beschreibung möglichst *problemorientiert* abzufassen, bei geringem fachspezifischen Vorwissen ist die Beschreibung *handlungsorientiert* zu halten (siehe Kasten 20). Die handlungsorientierte Aufgabenbeschreibung soll verhindern helfen, dass die beobachteten Benutzungsprobleme überwiegend aufgrund fehlenden Fachwissens zustande gekommen sind.

Im Unterschied zum aufgabenorientierten Benutzungstest sind beim induktiven Benutzungstest *mehrere* Benutzer als Softwaretester beteiligt. Die Benutzergruppe sollte möglichst *repräsentativ* für die Population der Endbenutzer sein. Die Auswahl der Benutzer kann z.B. *zufällig* aus dem poten-

tiellen oder aktuellen Benutzerkreis erfolgen (für weitere Stichprobentechniken siehe Bortz 1984, Kapitel 4). Die ausgewählten Benutzer sollten das zu testende System nicht kennen. Da sich diese Bedingung bei grösseren Entwicklungsphasen bzw. Weiterentwicklungen oftmals nicht einhalten lässt, muss die Vorerfahrung der Benutzer mit dem System bzw. ähnlichen Systemen kontrolliert werden. Die Gruppengrösse sollte aus statistischen Gründen zwischen sechs und zwanzig Personen liegen (zur Abschätzung der Effektgrösse und Testpersonenanzahl siehe Bortz 1984, Kapitel 6). Wenn kein oder nur sehr wenig Wissen über die Population der Endbenutzer vorliegt, lohnt es sich, die Benutzergruppe hinsichtlich der folgenden Parameter möglichst *heterogen* zusammenzusetzen: EDV-Vorerfahrung, Alter, Geschlecht, Ausbildung, Beruf, zu unterstützende Aufgaben.

Anzustreben ist eine den realen Einsatzbedingungen möglichst entsprechende Beobachtungsumgebung. Da es für die spätere Auswertung jedoch sehr wichtig ist, das Verhalten der einzelnen Benutzer sowie das entsprechende Systemverhalten auf Video, Tonband, 'screenrecording' usw. aufzuzeichnen, empfiehlt es sich, einen ruhigen, abgeschlossenen Raum mit entsprechendem Mobiliar zu benutzen. Stehen keine speziellen Aufzeichnungsgeräte zur Verfügung, sind für den Testleiter einfach auszufüllende Protokollbogen (z.B. mittels Strichlisten für vorgegebene Problem-, Fehlerkategorien usw.) sehr nützlich.

Der Testleiter sollte sich stets darum bemühen, jedem Benutzer das Gefühl der Wichtigkeit und Wertschätzung zu vermitteln. Die folgenden drei Aspekte sind für die Schaffung eines vertrauensvollen und für Kritik offenen Klimas hilfreich.

(1) Mit möglichst allgemein verständlichen Worten wird dem Benutzer Ziel und Zweck des Benutzungstests erläutert (z.B. Test eines Prototyps in einem frühen Entwicklungsstadium usw.).

(2) Es ist besonders wichtig, dem Benutzer verständlich zu machen, dass das System und *nicht* er als Benutzer getestet werden soll. Jede Art von Schwierigkeiten seitens des Benutzers bei der Aufgabenbearbeitung sind von besonderem Interesse!

(3) Jedem Benutzer muss zugesichert werden, dass er die Durchführung des Benutzungstests jederzeit unterbrechen und sogar abbrechen kann, ohne dass irgendwelche negativen Konsequenzen (z.B. Wegfall einer zugesagten Bezahlung usw.) erfolgen. Die vollständige Freiwilligkeit

der Teilnahme und Zusicherung der Vertraulichkeit ist unabdingbar für die Gewinnung von brauchbaren Beobachtungen.

Jedem Benutzer werden alle für die Durchführung des Benutzungstests in dem Beobachtungsraum vorhandenen Geräte (Computer, Videokamera, Mikrophone usw.) hinsichtlich Einsatzzweck erläutert. Als besonders wichtig erweist sich eine möglichst standardisierte Einführung in die Handhabungs- und Bedienungsweise der zu testenden Software. Je nach Vorerfahrung des Benutzers können unterschiedliche Schwerpunkte gesetzt werden. Um die Motivation der Benutzer bei einer längeren Einführungsphase (in die Benutzung der Software) aufrechtzuerhalten, hat es sich als sinnvoll erwiesen, den Benutzer zur Generierung von Frage- und Problemstellungen über die 'Welt der Anwendungsobjekte' zu stimulieren und zur selbständigen Exploration anzuregen. Dennoch muss unbedingt darauf geachtet werden, dass jeder Benutzer das gleiche Vorwissen erhält. Da diese Bedingung nur schwer zu kontrollieren ist, begnügt man sich oftmals mit Personen *ohne* jegliches EDV-Vorwissen.

Um die Handlungsziele der Benutzer während der Aufgabenbearbeitung erfassen zu können, werden sie gebeten, 'laut zu denken' (siehe zur Methode Ericsson und Simon 1984, Moll 1987). Viele Benutzer haben jedoch Schwierigkeiten, ihre Gedanken während der Bearbeitung einer Aufgabe laut zu äussern. Es empfiehlt sich daher, dem Benutzer zu verdeutlichen, was mit 'lautem Denken' gemeint ist, und mit ihm einige Übungsbeispiele gemeinsam durchzugehen. Trotzdem neigen Benutzer einerseits bei hochroutinisierten Handlungen, andererseits bei komplexen Problemlösungsprozessen dazu, das 'laute Denken' einzustellen. Dies kann man durch folgende Massnahmen umgehen:

(1) Der Testleiter fordert den Benutzer in solchen Situationen zum 'lauten Denken' auf.

(2) Es werden *zwei* Benutzer als Paar mit der Aufgabenbearbeitung betraut, wodurch die verbale Kommunikation zwischen beiden Partnern das 'laute Denken' ersetzt.

(3) Man führt nach Beendigung des Benutzungstests dem Benutzer problematische Ausschnitte per Videoaufzeichnungen vor und bespricht mit ihm seine jeweiligen Handlungsabsichten und die dabei aufgetretenen Probleme (die Videokonfrontationsmethode, siehe Mittenecker 1987).

Es empfiehlt sich, vor Beginn einer Testserie ein bis zwei Probedurchläufe zur Optimierung der gesamten Testprozedur durchzuführen. Als bedeutsa-

me Einflussgrössen auf die Art und Weise der Aufgabenbearbeitung haben sich die EDV-Vorerfahrung der Benutzer und die Hilfestellungen durch den Testleiter herausgestellt. Beide Variablen sollten erfasst werden, um sie später bei der statistischen Auswertung berücksichtigen zu können. Die Vorerfahrung lässt sich mittels Fragebogen erfassen. Die Art und die Anzahl der gegebenen Hilfestellungen des Testleiters wird von diesem während des Benutzungstest auf einem Protokollbogen vermerkt.

Als Testkriterien werden alle erhobenen Messwerte bezeichnet, welche bei der Auswertung Aufschluss über die Güte der Benutzbarkeit des zu testenden Systems Auskunft geben können. Die Menge der Testkriterien teilt sich auf in die Menge der *qualitativen* Messgrössen (problematische Benutzungssituationen, Fehlerarten usw.) und die Menge der *quantitativen* Messgrössen auf (Bearbeitungsdauer, Anlernzeit, Überlegungszeit, Anzahl Tastendrucke, Anzahl Fehler; siehe Dumas und Redish 1993).

Für die Gestaltung des zeitlichen Rahmens gibt es grundsätzlich zwei verschiedene Vorgehensweisen:

(1) Der Benutzer wird aufgefordert, die gestellte Aufgabe solange zu bearbeiten, bis er sie vollständig gelöst hat oder die Bearbeitung auf eigenen Wunsch abbricht.

(2) Dem Benutzer wird eine maximale Zeitdauer für die Aufgabenbearbeitung zur Verfügung gestellt.

Falls die Aufgabenbearbeitungszeit als ein relevanter Indikator für Benutzungseigenschaften des zu testenden Systems herangezogen werden soll, empfiehlt sich die erste Variante. Bei der zweiten Variante muss man für die Auswertung den erreichten Lösungsgrad der Aufgabe bewerten. Diese Bewertung ist oft nicht einfach möglich.

Der Testleiter hält sich während der Aufgabenbearbeitung ruhig im Hintergrund. Für ihn ist es sehr wichtig, seinen Wunsch, dem Benutzer in problematischen Situationen sofort zu helfen, zurückzuhalten. Hier kann eine klare Absprache zwischen Testleiter und Benutzer hilfreich sein: Nur wenn sich der Benutzer explizit an den Testleiter wendet und um Hilfe nachfragt, wird der Testleiter aktiv. Ein zu frühes Eingreifen des Testleiters hindert den Benutzer, eine eigene Lösung zu finden. Als Testleiter sollten daher keine an der Entwicklung des zu testenden Systems unmittelbar beteiligten Personen eingesetzt werden. Es wird sogar verschiedentlich empfohlen, um eine möglichst realistische Beobachtungssituation herzustellen, dass der Testleiter sich völlig passiv verhält und dies dem Benutzer vorher deutlich

macht. Weitere wertvolle Hinweise zum Testleiterverhalten findet man in Gniech (1976, S. 41-57).

Jedem Benutzer muss die Gelegenheit gegeben werden, sofern dies nicht schon im Rahmen der Videokonfrontation eingeplant ist, für ihn wichtige und noch offene Fragen zu Zwecken und Zielen des Benutzungstests, problematische Situationen während der Aufgabenbearbeitung usw. in einem Gespräch nach Beendigung der Aufgabenbearbeitung mit dem Testleiter klären zu können. Meistens erhält man aus diesen Gesprächen sehr zutreffende Problemsichten. In dieser Nachbereitung lassen sich auch Fragebogen mit standardisierten oder offenen Fragen zur Beantwortung spezieller Benutzungsaspekte einsetzen.

Während der Durchführung eines Benutzungstests wird man immer wieder überraschende und sehr informative Benutzungsweisen ('critical incidents', 'Stolpersteine') beobachten können. Es lohnt sich, diese auf Video (am besten Super-VHS wegen der höheren Bildauflösung) aufzuzeichnen, um sie dann mit den Entwicklern später detailliert diskutieren zu können. Besser ist es jedoch oftmals, die Entwickler – oder einen Repräsentanten durch eine Einwegscheibe vom Testraum getrennt dem Test beiwohnen zu lassen. Wichtig ist dabei, dass die beobachtbaren Schwierigkeiten niemals dem Benutzer, sondern ausschliesslich der Benutzbarkeit des zu testenden Systems angelastet werden. Wenn mehrere Benutzer dieselben Schwierigkeiten hatten, kann es nicht an ihnen liegen!

Die aufgezeichneten Beobachtungsdaten (Super-VHS-Video, Tonband, Logfile, 'screenrecording') müssen hinsichtlich der designrelevanten Informationen ausgewertet werden. Dazu empfiehlt es sich, auf der Grundlage der durchgeführten Beobachtungen ein Auswertungsraster aufzustellen und dieses Raster systematisch auf die Beobachtungsdaten aller Benutzer anzuwenden. Aufwendig und schwierig sind die qualitativen Auswertungen von Logfiledaten, weil es sich hierbei oftmals um komplexe Mustererkennungsprozesse handelt, welche bisher nur begrenzt automatisch ausführbar sind. Mit dem Auswertungsprogramm AMME lassen sich einige relevante Messgrössen gewinnen (siehe hierzu Rauterberg 1993). Ein umfangreiches Messinventar zur Messung der Benutzungsfreundlichkeit wird vom National Physical Laboratory in England angeboten (MUSiC 1993).

Als eine besonders informative Quelle für designrelevante Entscheidungen hat sich die Analyse der Videoaufzeichnungen ergeben. Hierbei werden Erklärungsmodelle für bestimmtes Benutzungsverhalten aufgestellt, um das beobachtbare Verhalten in problematischen Situationen handlungspsycho-

logisch begründen zu können. Die eingehendere Beschäftigung mit diesen Situationen führt dann unter Berücksichtigung der oben beschriebenen Kriterien zu konstruktiven, tragfähigen Gestaltungsvorschlägen.

Um die in der qualitativen Auswertung gewonnenen Erklärungen für einzelne Benutzungsprobleme auf eine möglichst breite Basis zu stellen (Problem der 'Generalisierbarkeit'), sind statistische Auswertungen unumgänglich. Diese können von einfachen, deskriptiven Häufigkeitsauszählungen bestimmter Problemklassen bis hin zu komplexen inferenzstatistischen Zusammenhangsanalysen gehen. Dazu ist es notwendig, dass die verschiedenen Benutzungseigenschaften (Dauer der Aufgabenbearbeitung, mittlere Überlegungszeit, Fehlerart, Ausmass an Beanspruchung usw.) pro Benutzer in quantifizierter Form vorliegen.

4.14.3 Deduktiver Benutzungstest

Deduktive Benutzungstests dienen primär der Entscheidungsfindung zwischen verschiedenen Systemalternativen bzw. zur Kontrolle der erreichten Verbesserung und erst sekundär der Gewinnung von Gestaltungsvorschlägen. Es ergeben sich daher einige Unterschiede in den Anforderungen an die Durchführung von deduktiven Benutzungstests.

Im Gegensatz zu induktiven Benutzungstests müssen die verschiedenen zu testenden, alternativen Systemversionen beim deduktiven Benutzungstest verstärkt der Forderung nach einem – an realen Einsatzbedingungen gemessenen – realistischen Systemverhalten (das heisst, möglichst funktionale Vollständigkeit, adäquates Systemantwortzeitverhalten usw.) genügen. Diese Anforderungen sind deshalb wichtig, weil die Entscheidung zwischen den Systemalternativen primär mittels quantitativer Messgrössen gefällt wird.

Je nachdem, ob viele Benutzer wenig Zeit haben (Fall I) oder ob wenige Benutzer eher mehr Zeit haben (Fall II), um an einem Benutzungstest teilzunehmen, ergibt sich ein unterschiedliches Testdesign. Im Fall I wird pro Systemalternative eine eigene Gruppe gebildet. Im Fall II hat lediglich eine Gruppe alle Systemalternativen zu testen. Um im Fall I jedoch Lerneffekte zu kontrollieren, muss die Reihenfolge der zu testenden Systeme innerhalb dieser einen Gruppe ausbalanciert werden. Im Fall II sollten dann auch für jede Systemalternative *parallele* Testaufgaben erstellt werden.

Die Gruppengrösse sollte aus statistischen Gründen mindestens sechs Personen pro Gruppe betragen. Wenn man plausible Annahmen über den zu

erwartenden Oberflächeneffekt hinsichtlich einer quantitativ zu messenden Benutzungseigenschaft (z.B. Dauer der Aufgabenbearbeitung, Anzahl Fehler usw.) für die Alternativen hat, kann man die genau benötigte Gruppengrösse auch ausrechnen. Die einzelnen Gruppen müssen hinsichtlich verschiedener Parameter möglichst homogen sein: EDV-Vorerfahrung, Geschlecht, Alter, Beruf. Mittels inferenzstatistischer Auswertungsverfahren können die gewonnenen Beobachtungsergebnisse auf ihre *Generalisierbarkeit* hin getestet werden.

4.15 SCHULUNG

Im folgenden Abschnitt geht es um die Schulung der Benutzer im Umgang mit dem neuen EDV-System.

Warum

Die Schulung der Benutzer wird in der Praxis leider oft vernachlässigt; es genügt keinesfalls, die Mitarbeiter mit einem oder mehreren Handbüchern ihrem Schicksal zu überlassen! Die Folgen sind in der Regel überforderte und frustrierte Benutzer; ursprünglich vorhandene positive Einstellungen zu Computersystemen können sich in Ablehnung und Abneigung umwandeln, Ängste vor Bedienungsfehlern können entstehen, letztlich kann die Systembenutzung überhaupt verweigert werden. Um diese Folgen zu vermeiden und den Benutzern eine effiziente Beherrschung der Systemfunktionalität zu ermöglichen, ist eine seriöse Schulung unumgänglich!

Wer

Grundsätzlich sollen *alle* Benutzer durch eine gründliche Ausbildung auf das neue System vorbereitet werden. Das heisst, dass auch gelegentliche Benutzer – wie z.B. Vorgesetzte, die das System nur einmal wöchentlich zur Abfrage von Statistiken benutzen – zumindest minimale Kenntnisse über den Umgang mit dem System erhalten sollten; sie werden dadurch unabhängiger von Hilfeleistungen geübter Benutzer.

Was

Je besser die Ausbildung, desto besser ist auch die Nutzung der Systempotentiale! Die Vermittlung von *Bedienungskenntnissen* (sogenannte *operative Kenntnisse*) *allein* ist *ungenügend!* Es hat sich als sehr wichtig gezeigt, dass den Benutzern zusätzlich auch *funktionale Kenntnisse* über den Computer und die Software vermittelt werden. Erst diese Kenntnisse ermögli-

chen den Benutzern, sich ein realistisches gedankliches Modell – das heisst, ein vorstellungsmässiges Abbild – der Funktionen und Zusammenhänge des Systems aufzubauen.

Beispielsweise wird der Umgang mit dem Computer erleichtert, wenn bekannt ist, was Arbeitsspeicher, Betriebssystem, Directory, Festplatte usw. sind. Dadurch verhilft man den zukünftigen Benutzern zur umfassenden Beherrschung des neuen Arbeitsmittels, was ein Verstehen der (z.B. durch Tastendruck ausgelösten) Verarbeitungsprozesse einschliesst. Das Gefühl der Unterlegenheit gegenüber einer nicht durchschaubaren Maschine sowie die Angst vor fehlerhaften Eingaben und deren Auswirkungen können damit weitgehend abgebaut werden. Gute gedankliche Modelle erleichtern darüber hinaus auch die Umstellung bei späteren Erweiterungen oder Systemwechseln, da die neu zu lernenden operativen Kenntnisse sich bei ähnlichen Funktionen besser aneignen lassen.

Inhalt der Schulung:

(1) Grundlagen der EDV,

(2) das EDV-System im Betrieb: Ebenen, Bereiche, Komponenten und deren Funktionen,

(3) funktionale Kenntnisse: Funktionen der Software und deren Einsatz im Arbeitsablauf, Unterschiede zur früheren Arbeitsweise,

(4) operative Kenntnisse: Bedienung der Geräte und Programme, Verhalten bei Fehlern, Systemabstürzen,

(5) Benutzung von Handbüchern, On-line-Tutorials.

Wie

Die erste Schulung findet vorzugsweise in Form eines ein- bis mehrtägigen, betriebsinternen Kurses mit Übungsmöglichkeiten am Bildschirm statt. Eine Mischung von Theorie mit eingestreuten praktischen Übungen des Gelernten hat sich bewährt. On-line-Tutorials können der individuellen Vertiefung – im Selbststudium – und Einübung dienen, sollten aber keinen Ersatz für 'traditionelle' Schulung mit einem Lehrer bilden, da bei Unklarheiten und Rückfragen ein Lehrer bessere Erklärungsmöglichkeiten hat (siehe auch Abschnitt 'Übungsarbeitsplatz'). Das Lernen in Kleingruppen mit gegenseitigen Unterstützungsmöglichkeiten ist empfehlenswert. Zwei Aspekte sind bei der Schulung besonders zu berücksichtigen:

(1) Aufgabenbezug
Das Lernen sollte im Kontext der konkreten, eigenen Aufgaben und Arbeitsabläufe erfolgen; dadurch werden die Einsatzmöglichkeiten der Software und deren Nutzen bei der Aufgabenerfüllung besser erkennbar. Ausserdem wird das Lernen von überflüssigem Ballast vermieden. Dies kann zur Folge haben, dass für Mitarbeiter verschiedener Abteilungen oder mit unterschiedlichen Aufgaben (z.B. Sachbearbeiter, Sekretär, Vorgesetzter) auch verschiedene Kurse mit dementsprechenden Anwendungsschwerpunkten durchgeführt werden. Aufgabenorientierte Schulung beinhaltet ein Ausrichten der Ausbildungsinhalte und Übungen auf die spezifischen Aufgaben einer Mitarbeitergruppe oder Organisationseinheit. Dies setzt voraus, dass auch die Ausbilder nicht nur über Systemkenntnisse, sondern auch über die jeweiligen Fachkenntnisse verfügen.
Der Erfolg derartiger Kurse rechtfertigt ihren vergleichsweise hohen Aufwand! In externen Kursen ist dieser Bezug auf die konkreten Aufgaben kaum möglich; ihr Wert für die Vermittlung von allgemeinem Grundlagenwissen über den Umgang mit einem Programm für Anfänger oder Spezialkenntnissen zur Ausreizung des Systempotentials für versierte Benutzer sei hier keineswegs angezweifelt. Betriebliche Unterstützung bei der Umsetzung des Gelernten auf die eigene tägliche Arbeit erweist sich aber in der Regel für viele Benutzer als hilfreich! Ferner besteht bei einigen Anbietern von externen Kursen die Möglichkeit der Abstimmung von Standardkursen auf betriebliche Bedürfnisse, sofern die ganze Klasse aus Mitgliedern der Belegschaft besteht.

(2) Den Kenntnissen angepasste Schulung
Um Überforderung der einen Teilnehmer und Unterforderung anderer Teilnehmer zu vermeiden, sollte die Schulung dem Ausbildungsstand und den Erfahrungen der Teilnehmer angepasst sein. Analog zu einer Skischule sollten die Auszubildenden in Gruppen von Teilnehmern mit gleichen bzw. ähnlichen Kenntnissen und Vorerfahrungen eingeteilt werden. Damit kann der Lernfortschritt für die meisten Teilnehmer optimal gefördert werden.

Übungsarbeitsplatz:

Zum Zwecke der spielerischen und damit angstfreien Gewöhnung an ein neues technisches System hat sich in der Praxis folgendes Vorgehen bewährt: Einige Zeit vor der Einführung wird ein Übungsarbeitsplatz eingerichtet, der ein Abbild der geplanten Arbeitsplätze darstellt, dessen techni-

sches System zur Vermeidung nachteiliger Auswirkungen infolge Fehlbedienung aber mit simuliertem Datenmaterial arbeitet. Den Mitarbeitern wird – nach einer angemessenen Schulung – freier Zugang zu diesem neuen Arbeitsplatz gewährt, so dass sie selbständig die Handhabung und Möglichkeiten des neuen Arbeitsmittels spielerisch erkunden können, da sie ja bei Fehlern keine gravierenden Folgen zu befürchten haben. Ein weiterer Vorteil dieses Vorgehens besteht in der frühzeitigen Entdeckung eventuell vorhandener Nachteile des neuen Arbeitsmittels, wodurch Verbesserungsvorschläge in der Realisationsphase berücksichtigt werden können.

Kasten 21: Beispiel Schneeballprinzip und mögliche Gefahren.

> Drei Monate vor der Einführung eines Informationsabfragesystems in einer Versicherung wurde die Ausbildung der Mitarbeiter, die nach dem *Schneeballprinzip* erfolgen sollte, in Angriff genommen. Da gerade Ferienzeit war (Personalengpass, viele Diebstähle, Unfälle), mussten die Agenturen die am ehesten abkömmlichen Mitarbeiter delegieren. Diese wurden in einem halbtägigen Kurs rezeptartig mit der Bedienung des Bildschirmgerätes und den wichtigsten Abfrageprozessen vertraut gemacht. Da es noch keine Terminals gab, unterrichteten sie anschliessend ihre Arbeitskollegen anhand von Folienmaterial. Als das System dann etwa zweieinhalb Monate später eingeführt wurde, konnte kaum ein Mitarbeiter damit umgehen. Erst durch intensives Selbststudium des Benutzerhandbuches konnte die Unterstützung des Systems allmählich genutzt werden; einige Mitarbeiter waren aber schon so frustriert von ihren erfolglosen Bemühungen, dass sie die Arbeit mit dem Bildschirm vermieden und nach wie vor mit vorhandenen Archivinformationen arbeiteten.

'Schneeballprinzip':

Vielfach erfolgt die Schulung in Betrieben nach dem sogenannten '*Schneeballprinzip*'; nur ein kleiner Teil der Mitarbeiter besucht die Schulung und vermittelt nachher das erlangte Wissen und Können in der Arbeitsgruppe oder Abteilung selber weiter an die übrigen Mitarbeiter (siehe Kasten 21).

Dieses Beispiel zeigt u.a., dass das Schneeballprinzip eine sorgfältige Auswahl der zuerst Auszubildenden aufgrund ihrer bisherigen Erfahrung und weiteren Arbeit mit Computern, ihrer Motivation und didaktischen Fähig-

keiten sowie auch genügend Zeit erfordert. Eine vorangehende didaktische Schulung erweist sich vielfach als notwendig. Häufig werden diese Ausbilder dann auch aufgrund ihrer ausgeprägten Systemkenntnisse neben ihrer eigentlichen Aufgabe auch zu Koordinatoren bzw. Kontaktpersonen zwischen EDV- und Fachabteilung ernannt und bilden so eine dezentrale Anlaufstelle – lokale Experten – für Benutzer, die Fragen oder Schwierigkeiten im Umgang mit dem System haben. Ihre Rolle ist so wichtig, dass sie gegebenenfalls durch eine Entlastung von Arbeitsaufgaben unterstützt werden sollte, um ohne Überbelastung seriös wahrgenommen werden zu können.

Neben traditionellen Formen der Ausbildung hat sich gezeigt, dass das sogenannte 'exploratorische Lernen', bei welchem Benutzern die Möglichkeit zur eigenständigen Erkundung eines Systems mit einem Minimum an vorgegebenen Informationen geboten wird, zu guten Ergebnissen führt (siehe Dutke 1994).

Wann

Die Schulung muss mit der Systemeinführung zeitlich so koordiniert werden, dass das Gelernte laufend angewendet bzw. geübt werden kann. Erfolgt die Schulung – wie im Beispiel 'Schneeballprinzip' – zu früh, so gehen die operativen Kenntnisse infolge Nichtanwendung verloren; allenfalls können theoretische Grundbegriffe und funktionale Kenntnisse einige Zeit vor der Einführung vermittelt werden. Neben der Schulung in Kursen sollte für die Benutzer auch Literatur über das System wie z.B. verständliche, didaktisch aufbereitete Handbücher u.ä. verfügbar sein, damit sie im Selbststudium ihr Wissen erweitern bzw. vertiefen oder Gelerntes – aber bis anhin nicht Gebrauchtes – wiederholen können. Während der Einführungsphase sollte auch jederzeit ein Experte zumindest telefonisch erreichbar sein; in der Regel bevorzugen Benutzer gerade in dieser Phase menschliche Unterstützung gegenüber einer Unterstützung durch Handbücher, Hilfesysteme und Lernprogramme, weil ein Experte besser kontextbezogene, auf individuelle Bedürfnisse abgestimmte Hilfe leisten kann.

Eine seriöse Schulung bildet nicht nur die Basis für eine effiziente Systembenutzung; sie beeinflusst auch die Einstellung und das Verhalten der Benutzer bei zukünftigen Systemerweiterungen oder -veränderungen.

4.16 NACHEVALUATION BEIM SYSTEMEINSATZ

Mit der Einführung des Systems bzw. der Überführung in den produktiven Betrieb sind die Aufgaben der Systementwicklung noch nicht abgeschlossen, wie fälschlicherweise oft geglaubt wird. Erst die Erfahrungen im Alltagsbetrieb lassen eine endgültige Beurteilung zu, ob die angestrebten Ziele erreicht wurden bzw. welche Mängel – z.B. Details der Benutzungsoberfläche – noch vorhanden sind und korrigiert werden müssen. Ausführliche, gewissenhaft durchgeführte Programmtests vor der Systemeinführung sind zwar unabdingbar, können aber für einen Teil aller Geschäftsfälle höchstens logische Fehler oder Plausibilitätsfehler der Software ermitteln. Jedoch für die überall auftretenden und schwer vorhersehbaren Sonderfälle im Alltagsbetrieb zeigen sich eventuell noch vorhandene Schwachstellen und Defizite der Software! Deshalb ist eine Nachevaluation zirka drei Monate (die Dauer ist abhängig vom konkreten Projekt) nach der Systemeinführung angezeigt. Sie soll darüber Aufschluss geben,

(1) ob allgemein die gesetzten Ziele erreicht wurden bzw. weshalb nicht,

(2) ob die Software noch Fehler, z.B. in der Programmlogik, aufweist,

(3) inwieweit die angebotene Unterstützung von den Benutzern auch beansprucht wird bzw. ob wesentliche Systemfunktionen von den Benutzern – sei es mangels Kenntnis, fehlender Aufgabenangemessenheit oder infolge von Handhabungsproblemen – nicht genutzt werden, welche Mängel der Benutzungsoberfläche Benutzungsprobleme verursachen und

(4) ob für die vollumfängliche Nutzung der Systempotentiale ein zusätzlicher Schulungsbedarf besteht.

Methodisch gesehen bestehen folgende Möglichkeiten für eine derartige Nachevaluation:

(1) Messung des zeitlichen Aufwandes für einzelne Bearbeitungsgänge oder ganze Abläufe.

(2) Messung der Anzahl bearbeiteter Fälle im Vergleich mit der Situation vorher.

(3) Messung der Anzahl aufgetretener Fehler; qualitative Analyse der Fehler: Welche Fehler? Immer die gleichen Fehler?

(4) Protokollierung von Dialogabläufen mittels Logfiles, aus Gründen des Datenschutzes aber ohne die Erhebung persönlicher Benutzerdaten!

Derartige Protokolle können hilfreiche Hinweise für Schwachstellen – Stolpersteine – im Dialog geben; der Auswertungsaufwand kann allerdings beträchtlich sein!

(5) Persönliche Interviews mit und/oder schriftliche Befragung von Benutzern.

(6) Protokolle der Hotline oder einer sonstigen Supportstelle: Welche Fragen werden häufig gestellt, wo treten die meisten Probleme auf?

Anhand der Untersuchungsergebnisse wird dann zu entscheiden sein, ob und gegebenenfalls welche Änderungen, Korrekturen in der Software nötig sind bzw. ob ganze Module neu zu entwickeln sind. Sinnvollerweise sollten diese Änderungen vom Entwicklungsteam selbst und, weil der Einarbeitungsaufwand gross ist, *nicht* von einer generell mit Programmwartungsaufgaben beauftragten Gruppe durchgeführt werden. Die realisierten Änderungen sollten dann auch den Benutzern – on-line oder schriftlich – mitgeteilt werden. Erst nach einer derartigen Nachevaluation werden die Systemeffekte abschliessend beurteilbar sein. Der Einsatz wird so lange reibungslos funktionieren, bis aufgrund eines erhöhten Anspruchsniveaus der Benutzer oder eines gestiegenen Bedürfnisses nach zusätzlicher Unterstützung eine Weiterentwicklung erwartet wird.

5 Fazit

In diesem Teil B wurden eine Reihe von Methoden, Techniken und Verfahren vorgestellt, welche jeweils auf die eine oder andere Art eine Beteiligung von Benutzern ermöglichen. Für weiterführende Darstellungen zu diesen Methoden sei die angegebene Literatur empfohlen. Hauptanliegen dieses Abschnittes ist die Vermittlung grundlegenden Wissens zur Auswahl einzelner Methoden. Dabei haben wir das Schwergewicht auf die frühen Phasen (Quadrant-I und -II) gelegt. Fundiertes Wissen zur Anwendung dieser Methoden und Moderationstechniken wird in der angegebenen Literatur und entsprechenden Trainings- bzw. Schulungskursen vermittelt.

TEIL C: FALLBEISPIELE

Vorbemerkung zu den Fällen

In den folgenden fünf Fallbeispielen werden wir aufzeigen, wie sich die einzelnen Beteiligungsmethoden je nach Projekttyp praktisch einsetzen lassen, welche Ergebnisse dabei erzielt wurden und welche Probleme sich dabei im einzelnen ergeben haben.

Die geschilderten Fälle sind nicht erfunden, sondern stammen aus der Praxis! Aus didaktischen Gründen wurden sie vereinfacht. Besonders interessante Aspekte wurden hervorgehoben. Die Fälle sollen die in den bisherigen Kapiteln dargestellten Konzepte veranschaulichen. Sie zeigen auch die in der Praxis möglichen, vielfältigen Kombinationen von Formen, Vorgehensweisen und Methoden partizipativer Softwareentwicklung. Die Fälle können auch in Kursen für Übungszwecke verwendet werden.

Rahmen-bedingungen	Fall 1: Versicherung	Fall 2: KBT	Fall 3: Bibliothek	Fall 4: ADI	Fall 5: PPS
Projekttyp	A	A	A	D	A
Projekt-komplexität	hoch	hoch	mittel	mittel	sehr hoch
Projekt-innovativität	gering	hoch	mittel	gering	mittel
Benutzer-gruppen	heterogen	heterogen	heterogen	sehr heterogen	heterogen
Aufgaben	anspruchsvoll	heterogen	Routine-aufgaben	unbekannt	anspruchsvoll

Zitate ("...") sind wörtliche Aussagen beteiligter Personen.

Fall 1: Versicherung

1 AUSGANGSLAGE

In einer grossen Versicherung wurde der Beschluss zu einer Totalrevision des bestehenden, zirka 12 Jahre alten EDV-Systems gefasst. Die Revision sollte vor allem folgende Aspekte umfassen:

(1) Es sollten beträchtlich mehr aktuelle, fehlerfreie Informationen für tägliche Bearbeitungsvorgänge und für statistische Auswertungen verfügbar sein.

(2) Das System musste eine neue, benutzungsfreundliche Oberfläche aufweisen; dabei war insbesondere der Erweiterung von Suchmöglichkeiten der Benutzer nach Informationen anhand verschiedener Kriterien gebührend Rechnung zu tragen.

Das System sollte einem 'breiten' Benutzerkreis (Vorgesetzte unterschiedlicher Stufen; Sachbearbeiter; Sekretärinnen; angelernte Bürohilfskräfte) mit stark unterschiedlicher Computererfahrung und Benutzungshäufigkeit des Systems ermöglichen, mit geringem Aufwand umfassende Informationen über einen Geschäftsvorfall oder gezielt einzelne (Detail-)Informationen abzurufen; ausserdem sollte es zur Erfassung neuer Daten dienen. Das gleiche System sollte auch in verschiedenen Geschäftsbereichen (Departemente wie Policen, Schaden, Marketing usw.) mit dementsprechend unterschiedlichen Aufgaben sowie in Niederlassungen mit unterschiedlicher Organisation als technisches Hilfsmittel dienen. Die Heterogenität des Benutzerkreises, der Aufgaben und der Organisation erforderte also ein äusserst flexibel adaptierbares System, das in Zukunft rund 2500 Benutzern für zirka 80'000 Abfragen und 1000 Erfassungen pro Tag zur Verfügung stehen sollte. Obwohl das Vorhaben nicht sehr innovativ oder besonders komplex war, musste das System von Grund auf neu entwickelt werden, da wegen der genannten Anforderungen an eine Revision des bestehenden Systems nicht mehr zu denken war.

2 PROJEKTORGANISATION

Mit der Entwicklung des Systems wurde eine Projektgruppe beauftragt, die sich aus einem Analytiker (Projektleiter) und zwei Programmierern zusammensetzte; ergänzt wurde diese Gruppe durch einen internen Berater (Be-

nutzer-EDV-Koordinator mit langjähriger Erfahrung) sowie durch einen externen Berater für Fragen der Softwareergonomie und der Benutzerbeteiligung. Es wurde auch eine Benutzergruppe (N=11) gegründet, die sich stellvertretend für alle Benutzer an der Systementwicklung beteiligen sollte; die Aufgaben dieser Gruppe bestanden vorwiegend in der Formulierung von Anforderungen, der Beurteilung von Konzepten der Entwickler (im Sinne einer Vernehmlassung), der Weitervermittlung der Konzepte an die Arbeitskollegen und damit verbunden im Einholen und Weiterleiten von Feedback an die Entwickler.

Die Zusammensetzung der Benutzergruppe wurde im Hinblick auf die Repräsentativität der vertretenen Anwendungsbereiche bestimmt; zur Hauptsache waren Sachbearbeiter mit Sonderfunktionen und erweiterten EDV-Qualifikationen in der Gruppe vertreten (tägliche Benutzer mit viel Erfahrung). Die Benutzergruppe hatte in allen – ausser technischen – Fragen der Anwendungsentwicklung Mitbestimmungsrecht und entsprechende Mitverantwortung. Ihre Entscheidungen konnten nur noch vom Steuerungs- und Kontrollausschuss, dem die Projektgruppe Rechenschaft über die Projektarbeit leisten musste, geändert werden. Das geschah aber in der Regel kaum.

3 PROJEKTABLAUF

In einer Vorstudie wurden Benutzeranforderungen erhoben, Arbeitsabläufe untersucht, Schwachstellen eruiert und generelle Ziele abgeleitet. Zur Erhebung der Benutzeranforderungen wurden die Abteilungsleiter aller Departemente, die Leiter der grösseren Niederlassungen sowie einige Leiter der zur Hauptsache betroffenen Bereiche mündlich oder schriftlich nach ihren Wünschen für das zukünftige System befragt; ausserdem erhielten die Benutzervertreter die Gelegenheit, in einem zweitägigen Workshop ihre Erfahrungen mit dem alten System zu diskutieren und ihre Anforderungen an das neue System zu formulieren. Als Ergebnis dieser Erhebungen ergab sich eine Liste mit 280 sehr detaillierten und z.T. auch widersprüchlichen Benutzerwünschen, deren Auswertung der Projektgruppe einige Mühe bereitete. Parallel dazu wurden in 13 Abteilungen am Hauptsitz sowie in zwei Niederlassungen (damit wurde die Repräsentativität der verschiedenen Bereiche sichergestellt) verschiedenartige Arbeitsabläufe (z.B. Adressenerfassung; Informationssuche bei der Schadenbearbeitung, im Marketing usw.) grob untersucht. Die Analyse dieser Abläufe zeigte einige Schwachstellen

und führte – zusammen mit den Benutzeranforderungen – zur Formulierung genereller Ziele seitens der Projektgruppe:

(1) Möglichkeit individueller Ausnutzung umfangreicher, aktueller, fehlerfreier Daten,

(2) Benutzerfreundlichkeit der Systemhandhabung,

(3) Anpassbarkeit an unterschiedliche Ablauforganisation, bereinigte, in grundlegenden Aspekten aber vereinheitlichte Abläufe,

(4) Kundenorientierung.

Auf dieser Grundlage wurden von der Projektgruppe sodann in zahlreichen Workshops generelle Lösungskonzepte entwickelt und in verschiedenen, 'inoffiziellen' Vernehmlassungen (Kollegen) präzisiert; unterschiedliche Varianten wurden Nutzwertanalysen unterzogen (wobei quantifizierte, qualitative Aspekte eine wesentliche Rolle spielten), mit Statistiken (wo verfügbar) belegt und mit einer Empfehlung für eine Variante in einem Bericht dem Kontrollausschuss vorgelegt. In der Gesamtwirtschaftlichkeitsrechnung wurden lediglich die Kosten beziffert; der Nutzen bzw. Ertrag wurde explizit nicht mit Produktivitätssteigerungen ("das Projekt dient nicht Rationalisierungszwecken"), sondern mit qualitativen Verbesserungen angegeben.

Das Problem der Anpassbarkeit des Systems an unterschiedliche organisatorische Abläufe wurde mit dem Konzept der flexiblen, ablauforganisatorischen Dezentralisierung, das sich durch Veränderung weniger Systemparameter direkt in den Niederlassungen verwirklichen lässt, gelöst.

Der Benutzerfreundlichkeit des Systems wurde grosse Bedeutung beigemessen. So sollten die Transparenz, Kompatibilität und Aufgabenangemessenheit des Systems durch den Benutzereinbezug bei der Informationsorganisation (Verteilung der Informationen auf Bilder) und der Gestaltung von Dialogabläufen sichergestellt werden. Dem heterogenen Benutzerkreis wird durch das Angebot von zwei völlig unterschiedlichen Interaktionsmöglichkeiten Rechnung getragen: Im sogenannten 'Standardmodus' wird der ungeübte, sporadische Benutzer über Menüs und Eingabemasken zu den gewünschten Informationen geführt; auch in diesem Modus sind aber selbst für Erfahrene noch genügend Freiheitsgrade vorhanden, so dass Menüs übersprungen und Bilder direkt aufgerufen werden können.

Im sogenannten 'Expertenmodus' erfolgt die Interaktion mit dem System über Kommandos mit Parametern; dabei können – im Sinne einer Indivi-

dualisierung – häufig benutzte Abläufe als Makro abgespeichert werden. Die Umschaltung zwischen den beiden Modi ist einfach (ein Tastendruck), theoretisch könnten Benutzer also mehrmals täglich wechseln. Dies kann z.B. bei selten benutzten Funktionen eine Erleichterung auch für 'Experten' darstellen. Der Flexibilität des Systems (im Sinne von Sprungmöglichkeiten innerhalb und auch zu anderen Anwendungen) sowie der Minimierung von Dialogschrittfolgen durch Direktzugriff und 'kurze Dialogwege' kam nämlich in den Benutzeranforderungen grosses Gewicht zu. Schnittstellen zu anderen Anwendungen sollten vereinheitlicht sowie grundlegende Elemente (wie Belegung von Funktionstasten, Anordnung von Menüs, Meldungen usw.) in Übereinstimmung mit anderen Anwendungen gestaltet werden, soweit dies sinnvoll erschien (Problem der Konsistenz zwischen alten und neuen Anwendungen).

4 WEITERER PROZESSVERLAUF

Die Projektgruppe befasste sich im weiteren – neben konzeptueller Arbeit an EDV-technischen Fragen – mit der Gestaltung einzelner Masken und stellte schliesslich fest, dass dieses Vorhaben ohne die vorherige Beantwortung übergeordneter Fragen (Funktionalität, Informationsorganisation usw.) zum Scheitern verurteilt war. Daraufhin wurde die begonnene Arbeit unterbrochen und die Bearbeitung der erwähnten, übergeordneten Aspekte in Angriff genommen.

An einer Tagung wurden den Benutzervertretern Vorschläge zum Dialogkonzept und zur Informationsorganisation präsentiert sowie erste Reaktionen eingeholt. Als 'Hausaufgabe' wurde den Teilnehmern aufgetragen, sich an ihrem Arbeitsort im Kreise der Kollegen gründlich und kritisch mit den vorgestellten Konzepten auseinanderzusetzen sowie Kritik und Verbesserungsvorschläge schriftlich und mündlich der Projektgruppe zurückzumelden, was auch geschah. Befürchtungen der Projektgruppe, dass die Benutzervertreter einzelne Teilkonzepte bzw. deren Darstellungsform nicht verstehen würden, erwiesen sich als unbegründet. Allerdings zeigte sich, dass die Integration der Teile zu einem Ganzen lediglich anhand schriftlicher Vorlagen eine Überforderung der Teilnehmer darstellte. Insgesamt verlief diese erste Zusammenkunft in entspannter Atmosphäre, die Teilnehmer zeigten sich motiviert, mehrheitlich gut qualifiziert und ihrer Verantwortung bewusst. Ihre Zustimmung zu Konzepten war der Startschuss für die unabänderliche Programmierung eines Systems, das in den nächsten ca. zehn Jahren für 2500 Benutzer eine Arbeitsgrundlage darstellt.

Für die weiteren Arbeitsschritte war eine aktivere Mitarbeit der Benutzergruppe im Entwicklungsprozess vorgesehen. Ihr Beitrag sollte sich nicht mehr 'nur' auf Beurteilungsleistungen und Korrekturvorschläge beschränken. Dementsprechend wurden in vier weiteren, ganztägigen Workshops von Benutzervertretern und Entwicklern Vorschläge der Benutzer zum Maskendesign, Konzepte für Erfassungsabläufe und das Gesamtsystem (Informationsabfrage, Erfassung) diskutiert und mit den Vorstellungen der Entwickler abgestimmt; in der Regel wurden die Ergebnisse dieser Arbeitstage von den Benutzervertretern an ihrer Arbeitsstelle im Kollegenkreis besprochen und allfällige Korrekturvorschläge bzw. anders gelagerte Vorstellungen an die Projektgruppe weitergeleitet, die sich ihrerseits mit den Konsequenzen für das weitere Vorgehen, in Angriff zu nehmende Aufgaben bzw. deren Delegation an eine Arbeitsgruppe (z.B. für die Formulargestaltung und für Druckoutput), befasste.

Nachdem schliesslich bezüglich Grob- und zum Teil auch Detailspezifikation des Systems zwischen allen Beteiligten grosse Übereinstimmung herrschte, konnte sich die Projektgruppe vermehrt der Programmierung zuwenden. Damit waren auch wesentliche Aufgaben der Benutzergruppe abgeschlossen; ein Teil der Benutzervertreter arbeitete allerdings noch bei den Programmtests mit und übernahm auch Ausbildungsaufgaben. Zirka drei Jahre nach Projektbeginn wurde das System dem produktiven Einsatz übergeben und erfüllt – zur Zufriedenheit der Benutzer – seine Unterstützungsfunktion zuverlässig; lediglich einzelne Details waren noch zu korrigieren.

5 FAZIT

Neben generellen Problemen wie Zeit- und Termindruck waren in diesem Projekt aber auch noch andere Probleme aufgetreten. So bezogen sich die Benutzeranforderungen im allgemeinen auf praktische Details, waren z.T. auch widersprüchlich und erschwerten damit der Projektgruppe eine inhaltliche Clusterung, eine Schwerpunktsetzung und die Überführung in Konzepte.

Die Benutzergruppe war zwar repräsentativ für die verschiedenen Anwendungsbereiche, aber nicht für verschiedene Benutzer-'Typen' (sporadischer, täglicher Benutzer; Manager, Sekretärin). Eine Erweiterung der Gruppe hätte aber den Kleingruppenrahmen zerstört (die Anzahl wäre auf über 20 Personen gestiegen) und auch Qualifikationsprobleme nach sich gezogen. Die Projektgruppe war deshalb angesichts dieser Nachteile und in Kenntnis der jetzigen Vertreter der Ansicht, dass die bestehende Zusam-

mensetzung die Berücksichtigung unterschiedlicher Bedürfnisse gewährleistete.

Die Benutzervertreter wurden für ihre Mitarbeit im Projekt nicht explizit – z.B. zu einem gewissen prozentualen Teil der Arbeitszeit – freigestellt; der Einbezug in die Softwareentwicklung war für sie eine Nebenbeschäftigung mit entsprechenden Folgen in belastungsmässiger Hinsicht. Ausserdem wurde der Projektgruppe von den betroffenen Vorgesetzten eine zeitliche Beanspruchung der Benutzervertreter von sieben Tagen zugestanden (die aber weit überschritten wurden!); offenbar waren sie nicht gewillt, einen Teil ihres Mitarbeiterpotentials übergeordneten Organisationszielen zu 'opfern'. Als weiteres Problem erwies sich eine – wenn auch nicht übermässige – (von Vorgesetzten manchmal angekündigt: "Wir schicken nächstes Mal einen anderen") – Fluktuation in der Benutzergruppe; eine Benutzergruppe sollte vom Beginn bis Projektende mehr oder weniger in ihrer Zusammensetzung bestehen bleiben, um die Konstanz der Arbeit nicht zu gefährden.

Ein wesentliches Problem der Entwicklung bestand in der Realisierung der Anpassungsfähigkeit des Systems an die unterschiedliche Ablauforganisation in den Niederlassungen (v.a. unterschiedliche Grade an Dezentralisierung von Aufgaben). Die Niederlassungen hätten sich einer Standardisierung von organisationalen Abläufen vehement widersetzt; deshalb wurde nach einer Lösung gesucht, die mittels Parametereinstellung am System eine flexible, das heisst, stufenartige Dezentralisierung ermöglicht und damit den Organisationseinheiten genügend Spielraum zur Gestaltung 'eigener' Organisationsformen lässt.

Das bezüglich Benutzerbeteiligung wohl interessanteste Problem zeigte sich an der ersten Benutzertagung: Den Benutzervertretern wurde – nach der Vermittlung von Grobkonzepten – der Datenkatalog für jedes Bild und anschliessend die Dialogstruktur (mittels Matrix- und Baumdarstellung) präsentiert; diese isolierte Betrachtung der beiden Teilaspekte konnte von den Teilnehmern problemlos nachvollzogen werden. Für den Benutzer ist aber das Wissen entscheidend, an welchem Ort im Dialog er welche Informationen zur Verfügung hat; die verlangte gedankliche Integration von Daten und Dialog aufgrund von Papierunterlagen erwies sich an der Tagung als zu komplexe Aufgabe für die Teilnehmer. Die Benutzervertreter mussten 'zu Hause' die Abbildung der Baumstruktur des Dialoges Schritt für Schritt durchgehen und bei jedem Kästchen der Abbildung das entsprechende Blatt mit dem Datenkatalog beiziehen. Dies wurde von ihnen zwar 'brav' ge-

schluckt, in der Tagungsbeurteilung – die sehr positiv ausfiel – wurde allerdings einhellig der Wunsch geäussert, die Daten und Dialoge auf dem Bildschirm sehen und damit experimentieren zu können, um sich ein anschauliches, realistisches Bild zu machen. Mit einem Dialog- und Maskengenerator wäre die Erstellung eines derartigen Prototyps in einem halben Tag möglich gewesen.

Der Projektleiter – ein qualifizierter Analytiker mit zirka zehn Jahren Erfahrung in Softwareentwicklung – hatte anfänglich grosse Bedenken bezüglich der Beteiligung von Benutzern; seine Argumente waren z.B.: "Ich kenne die Benutzer und ihre Bedürfnisse selbst. Die Benutzer wissen ja nicht, was sie wollen, oder können es nicht verständlich sagen. Die Benutzer sind sich selbst uneinig, das bringt ja nichts ausser Zeitverzögerung und Probleme. Man kann ja sowieso nicht alle einbeziehen." Im Laufe der Zusammenarbeit sind diese Bedenken aber einer sehr positiven Einstellung – " Das ist sehr wichtig. Die wissen vieles besser als wir" – gewichen. Man könnte sagen, er hatte Spass an der Zusammenarbeit mit der Benutzergruppe bekommen.

Im beschriebenen Fall ist die Beteiligung von Benutzern bei der Softwareentwicklung weder in Projekthandbüchern noch in sonstigen Reglementen festgelegt. Fortschrittliche Projektleiter führen Benutzerbeteiligung jedoch aus eigener Initiative durch.

Fall 2: Bankabteilung KBT

1 AUSGANGSLAGE

Aufgabe der Abteilung KBT (Konzernbetreuung) einer Bank ist die umfassende Betreuung und Koordination aller Geschäftsbeziehungen, die Konzerne und ihre Tochterfirmen mit allen in- und ausländischen Geschäftsstellen der Bank unterhalten. Die Abteilung ist in Gruppen von zwei bis fünf Angestellten aufgeteilt, die jeweils eine Gruppe von Konzernen betreuen. Wichtigste Grundlage einer erfolgreichen Betreuung ist die umfassende Information über den Kunden, seine Vorhaben und Bedürfnisse; dazu müssen über verschiedene Quellen (von Datenbasen bis zu Zeitungen) Informationen eingeholt und überprüft werden. Auf operativer Ebene müssen z.B. für die Vergabe von Krediten komplexe Modellrechnungen durchgeführt und Offerten erstellt werden. Dementsprechend reicht das Spektrum der Aufgaben von relativ unstrukturierten Teilaufgaben (z.B. Informationsgewinnung) bis zu präzise definierten administrativen Abläufen.

Eine Gruppe besteht in der Regel aus einem Chef, einem oder mehreren Assistenten, Sachbearbeitern und einer Sekretärin. Der Chef bemüht sich vor allem um den direkten Kontakt mit den Kunden und trifft die wesentlichen Entscheidungen, die von den Assistenten vorbereitet werden; sie sammeln die nötigen Informationen, prüfen sie und führen z.B. Modellrechnungen durch. Die Sekretärinnen sind hauptsächlich mit der Eingabe von Texten und Daten, der Anfertigung von Grafiken und der Telefonbetreuung beschäftigt.

Zur Bewältigung dieser Aufgaben stand den ca. 30 Mitarbeitern der Abteilung ein umfangreiches Arsenal technischer Arbeitsmittel als Unterstützung zur Verfügung: Bildschirmterminals zum Grosscomputer, PCs, Textverarbeitungsautomaten, Telex, Fax, programmierbare Taschenrechner usw. Diese Vielfalt von Einzelgeräten bewirkte aber auch viele Doppelspurigkeiten und Medienbrüche; so mussten z.B. gleiche Informationen, Formulare mehrfach parallel in verschiedenen Systemen gespeichert und bei Änderungen nachgeführt werden. Aufwendige administrative Prozeduren und das wachsende Geschäftsvolumen führten schliesslich zu einer permanenten Überlastung der Sachbearbeiter. Zur genaueren Analyse dieses unbefriedigenden Zustandes und seiner Ursachen wurde eine Vorstudie durchgeführt; ein Organisator und ein Informatiker studierten sechs Wochen lang

die bankfachlichen Aufgaben und Arbeitsabläufe in dieser Abteilung. Anschliessend wurden in einem Workshop mit Vertretern der Abteilung deren Probleme und Bedürfnisse nach technischer Unterstützung diskutiert.

Schliesslich wurde beschlossen, im Rahmen eines Pilotprojektes ein System zu entwickeln, welches das Management in den Geschäftsbeziehungen und bei der operativen Abwicklung von einzelnen Geschäftsfällen unterstützt. Dabei sollten "sämtliche EDV-unterstützbaren Funktionalitäten der Konzernbetreuung in einem homogenen System zusammengeführt werden". Das Vorhaben kann als komplex und innovativ bezeichnet werden; sowohl die Menge als auch die unterschiedliche Struktur der pro Kunde zu verarbeitenden Informationen wie auch deren komplexe Vernetzung stellten besondere Ansprüche. Für die Gestaltung der Benutzungsoberfläche wirkte der heterogene Benutzerkreis mit unterschiedlichen Aufgaben, Bedürfnissen und stark variierender Benutzungshäufigkeit erschwerend. Die infrastrukturellen Voraussetzungen für die Durchführung dieses Projektes können als sehr gut bezeichnet werden, da z.B. modernste Entwicklungstools verfügbar waren. Ausserdem mussten die Projektverantwortlichen die Entwicklung *nicht* nach dem betriebsüblichen Vorgehen (starres Phasenmodell) realisieren; vielmehr sollte eine neuartige Vorgehensweise (iterativ-zyklische Entwicklung unter Einsatz von Prototyping) erprobt werden. Der Komplexität des Vorhabens entsprechend sollte die Entwicklung unter Beteiligung der zukünftigen Benutzer erfolgen.

2 PROJEKTORGANISATION

Die Realisierung des Projektes wurde einer Projektgruppe übertragen, die sich aus vier Informatikern, zwei ständigen Benutzervertretern und einem Softwareergonomen der Bank zusammensetzte. Die Arbeit in dieser Gruppe war durch eine netzwerkartige Kooperation, häufige gemeinsame Diskussionen und Entscheidungsprozesse sowie durch vollständige, definierte Aufgabengebiete der einzelnen Mitglieder charakterisiert. Die Gruppe bzw. deren formeller Leiter musste einem aus Vertretern der EDV und der Fachabteilung gemischt zusammengesetzten Ausschuss Bericht erstatten.

Aufgaben der Benutzervertreter waren:

(1) Informationen der Projektgruppe (Stand der Arbeit, Vorschläge usw.) an die anderen zukünftigen Benutzer zu vermitteln und Feedback einzuholen, das der Projektgruppe für weitere Arbeiten diente,

(2) aktive Mitarbeit bei der Arbeitsanalyse, der Konzeption der Funktionsteilung sowie der Benutzungsoberfläche,

(3) Qualifizierung der Benutzer.

Ihre Mitarbeit beschränkte sich also nicht nur auf das Abgeben, Einholen von Informationen sowie die Beurteilung von Vorschlägen, sondern erstreckte sich auf aktive Konzeptions- und Entwurfsarbeit. Dies bedingte allerdings ihre zeitliche Freistellung für diese Aktivitäten; vor allem in den Anfangsphasen des Projektes (ca. ein Jahr!) arbeiteten sie zu 100% ihrer Arbeitszeit für das Projekt.

Neben diesen ständigen Benutzervertretern wurden fünf weitere Benutzer im Sinne von 'Reviewern' jeweils nach Bedarf zur Beurteilung von Konzepten und Prototypen, für Tests und zur Beantwortung, Diskussion offener Fragen der Projektgruppe beigezogen. Die übrigen Benutzer wurden an verschiedenen Informationsveranstaltungen mit Diskussionsmöglichkeit über den Stand des Projekts informiert und ihre Standpunkte dazu erfragt. Mit diesem Beteiligungsmodell sollte ein repräsentativer Benutzereinfluss gewährleistet und auch Effizienzkriterien bzw. zeitlichen Beschränkungen der Benutzer für die Mitarbeit Rechnung getragen werden. Die grundlegende 'Philosophie' der Projektgruppe – kennzeichnend für das gesamte Vorgehen – lautete: "Wir haben den Benutzern Dienstleistungen zu erbringen!"

3 PROJEKTABLAUF

In einer ersten Informationsveranstaltung wurden sämtliche Mitarbeiter der Abteilung über das geplante Projekt und seine Ziele, mögliche Vorgehensweisen und die angestrebte Benutzerbeteiligung in Grundzügen informiert und Fragen der Teilnehmer beantwortet. Das Projekt begann aber eigentlich damit, dass die Entwickler sich durch intensives Studium bankfachlicher Literatur, von Stellenbeschreibungen, Reglementen usw. mit den Aufgaben der Benutzer vertraut machten. Dies bildete auch eine gute Grundlage für die anschliessend mit einem Grossteil der Benutzer geführten Gespräche über ihre konkreten Aufgaben, Probleme und technischen Unterstützungsbedürfnisse. Auf dieser Basis skizzierten die Entwickler auf Papier einzelne Szenarien im Sinne einer ersten Diskussionsbasis und erstellten zu Demonstrationszwecken einen Prototyp (ohne Funktionalität!) für einen Teil eines Arbeitsablaufes. Vertieft wurde die Zusammenarbeit in vier halbtägigen Workshops mit jeweils etwa sieben Benutzern; bei dieser Gelegenheit wurde auch der Prototyp vorgestellt und diskutiert. Die schnelle Umset-

zung erster Ideen in ein anschauliches Verhandlungsobjekt zeigte bei den Benutzern enorme Motivationswirkung für die weitere Mitarbeit im Projekt!

Ausgehend von den Ergebnissen der im Abschnitt 1 erwähnten Vorstudie, den Gesprächen mit den Benutzern und den Workshops wurden die allgemeinen Zielvorstellungen für das System präzisiert. Das System sollte die Benutzer durch eine Kombination von Daten und – für die bankfachlichen Aufgaben – massgeschneiderten Bearbeitungsinstrumenten (Funktionen) unterstützen; die Ziele können im einzelnen wie folgt benannt werden:

(1) Entlastung der Benutzer von administrativen Vorgängen durch Automatisierung,

(2) Unterstützung bei der Suche, Selektion, Aufbereitung, Vernetzung und Verdichtung von Informationen,

(3) Unterstützung bei der Bearbeitung von Fällen durch integrierte Instrumente (Teilautomatisierung),

(4) Unterstützung bei der Präsentation von Informationen (Grafik).

Aus der Sicht der Entwickler betrachtet, sollte sich das System an einem erweiterten Dokumentenbegriff orientieren. "Ein Dokument wird aber nicht nur als eine statische Darstellung von Informationen aufgefasst. Vielmehr werden den Dokumenten zwei weitergehende Funktionalitäten zugeordnet:

- Zum einen eine dynamische Verarbeitungsleistung, die den Zugriff auf Daten (aus Datenbanken und anderen Dokumenten) und deren Prozessierung im Dokument selber erlaubt.
- Zum anderen ein Referenzierungsmechanismus, der die sinnvolle Vernetzung der Dokumente untereinander und damit ein schnelles Finden von zusammenhängenden – strukturierten und unstrukturierten – Informationen ermöglicht."

In iterativen Entwicklungszyklen wurden weitere Prototypen erstellt und in Tests von den Benutzern auf Benutzbarkeit und Funktionalität evaluiert. Die Resultate dieser Tests flossen jeweils in die nächste Entwicklungsstufe ein. Die wichtigsten Stationen dieser Entwicklung werden nachfolgend skizziert:

(1) Innerhalb von zwei Wochen wurde mit einem Hypertextsystem ein Prototyp (ohne Funktionalität) erstellt, der hauptsächlich der "Modellierung des Grundbildes der Desktop-Benutzungsoberfläche, der Ausarbeitung von Beispieldokumenten und Masken und insbesondere für das Auf-

zeigen der Navigationsmöglichkeiten im System (Wechsel zwischen verschiedenen Dokumenten und Masken)" diente. Dieser Prototyp bot eine gute Basis für Diskussionen mit den Benutzern über Gesamtzusammenhänge und Darstellungsaspekte. Parallel dazu wurden die Arbeitsabläufe genauer untersucht.

(2) Mit einem anderen, mächtigeren Hypertextsystem mit Programmiermöglichkeiten wurde ein weiterer Prototyp (mit Pseudofunktionalität) realisiert. Damit konnten schon relativ komplexe Dialogabläufe realistisch simuliert und Details der Benutzungsoberfläche dargestellt werden. Die Benutzer erhielten in verschiedenen Reviewsitzungen ein anschauliches Abbild des zukünftig möglichen Systems; sie konnten anhand dieses Prototyps schon weitgehend ein Urteil abgeben, inwieweit er bezüglich Darstellung, Dialog und Funktionalität ihren Anforderungen entspricht.

(3) Schliesslich wurden mit Werkzeugen der vierten Generation Prototypen mit echter Funktionalität erstellt, zum Produkt weiterentwickelt und teilweise in das definitive System integriert.

Neben der Verwendung von Prototypingwerkzeugen wurden für die Detailspezifikation auch strukturierte Analyse- und Spezifikationsmethoden (Datenflussanalyse, semantische Datenmodellierung) eingesetzt. Zusätzlich zu den beschriebenen Entwicklungsschritten musste auch noch die Einbettung des Systems in das Gesamtsystem realisiert werden, was vor allem eine Frage der technischen Schnittstellengestaltung bedeutete.

4 FAZIT

Der beschriebene Fall veranschaulicht eine Entwicklungsstrategie, die durch Prototyping eine schrittweise Präzisierung der komplexen Anforderungen und realistische Umsetzungsmöglichkeiten erzielt. Die rasche Erstellung von Prototypen schon am Anfang des Projektes erlaubte einen frühzeitigen Einbezug der Benutzer, was die Zusammenarbeit bei der Systementwicklung positiv beeinflusste; ausserdem konnten Missverständnisse in der Anforderungsdefinition und Konzeptformulierung vermieden bzw. frühzeitig genug geklärt werden. Im einzelnen ergaben sich durch den Einsatz von Prototyping folgende Vorteile:

(1) Die Kommunikation zwischen Entwicklern und Benutzern wurde durch das Vorliegen eines anschaulichen Diskussionsobjektes erleichtert.

(2) Die schnelle Realisierung von Vorschlägen der Benutzer wirkte motivierend für deren weiteres Engagement im Projekt; sie erkannten ihren Einfluss auf die Systemgestaltung.

(3) Softwareergonomische Gestaltungsaspekte konnten einfacher erprobt und überprüft werden.

(4) Den Entwicklern wurde die Strukturierung von Ideen sowie deren Umsetzung in verschiedene Lösungsvarianten erleichtert; sie hatten mehr Gestaltungsspielraum.

Von grosser Bedeutung für die Entwicklungsarbeit und die Kommunikation mit den Benutzern war auch das bankfachliche Wissen, das sich die Entwickler zu Projektbeginn angeeignet hatten. Ohne diese Kenntnisse hätte der Einsatz von Prototyping die geschilderten Vorteile kaum in vollem Umfang erbracht! Bevor man mit der Erstellung von Prototypen beginnt, muss man sich auch Klarheit darüber verschaffen, welche Fragen damit beantwortet bzw. welche Ziele damit verfolgt werden sollen. Prototyping erfordert ein gezieltes, hypothesengeleitetes Vorgehen; sonst besteht nämlich die Gefahr, dass es zur ziel- und planlosen Bastelei oder Spielerei ausartet! Ferner ist bei der Wahl der Werkzeuge auf geringen Aufwand für Einarbeitung und Erstellung von Prototypen zu achten, sonst kostet die Erstellung von Prototypen bald einmal soviel wie die Produktion des eigentlichen Systems. Gerade bei komplexen, innovativen Projekten dürften sich aber bei Beachtung dieser Gesichtspunkte die Vorteile von Prototyping in Zukunft vermehrt zeigen.

Fall 3: Bibliothek

1 AUSGANGSLAGE

Anfang der 80er Jahre beschloss die Leitung einer grösseren öffentlichen Bibliothek, den vorhandenen Mikrofichenkatalog durch einen computergestützten On-line-Katalog (OPAC; On-line Public Access Catalogue) zu ersetzen; der Katalog sollte über verschiedene öffentliche und private Netzwerke frei zugänglich sein und einen heterogenen Benutzerkreis (Studenten, Wissenschaftler, Bibliothekare von Instituten, Betrieben und anderen Bibliotheken, Private) bei ihren unterschiedlichen Aufgaben unterstützen. In diesem umfangreichen Projekt sollte also ein relativ komplexes, für damalige Begriffe innovatives System entwickelt werden.

Zur Realisierung des Vorhabens wurde eine Projektgruppe gebildet; sie bestand aus dem Leiter der EDV-Abteilung, der die Projektleitung übernahm sowie – je nach Phase und Bedarf – fünf bis neun Analytikern, Programmierern. Ausserdem wurden zwei Bibliothekare in die Entwicklungsarbeit einbezogen; ihre Aufgabe war es, die Benutzerbedürfnisse in die Entwicklung einzubringen und allgemein die Benutzerinteressen zu vertreten. Konkret bedeutete dies für sie die Erarbeitung von Vorschlägen zur Funktionalität des Systems sowie die Beurteilung von Konzepten der Entwickler bezüglich Benutzungsoberfläche und Funktionalität.

Einem Phasenmodell folgend entstand schliesslich durch die intensive Zusammenarbeit der Entwickler und der zwei Bibliothekare ein System, das mittels Telekommunikation Literaturrecherchen nach formalen und sachlichen Kriterien (inklusive verschiedener Hilfsfunktionen wie Verknüpfung von Begriffen, Fragmentsuche, Sortieren usw.), On-line-Bestellungen von Dokumenten sowie die Neuanmeldung eines Benutzers bzw. die Änderung von Benutzerdaten erlaubt. Sämtliche Funktionen (Recherchen, Bestellung, Ausleihe, Verwaltung der Benutzerdaten) können on line mittels Telekommunikation ablaufen und von den Benutzern kontrolliert werden.

2 UNTERSUCHUNG

Anlässlich des fünfjährigen Betriebes des On-line-Kataloges war es ein Anliegen der Bibliotheksleitung, auf dem Wege der empirischen Sozialforschung mehr über das Benutzerverhalten im Umgang mit diesem Informa-

tionssystem in Erfahrung zu bringen. Die Resultate einer Untersuchung sollten einerseits – im Sinne einer Momentaufnahme – die Beurteilung und Bewertung des Systems aus Benutzersicht wiedergeben, andererseits auch Ansätze für längerfristige Verbesserungsmöglichkeiten aufzeigen.

Mittels Interviews und einer Fragebogenerhebung bei 473 Benutzern wurde die Beurteilung des Systems, speziell der Benutzungsoberfläche, durch die Benutzer hinsichtlich verschiedener Aspekte (Maskendesign, Dialogsteuerung, Sprache, Abkürzungen, Fehlermeldungen usw.) erfasst. Ausserdem sollte die Untersuchung darüber Aufschluss geben, welche Abfragearten und Funktionen wie häufig benutzt werden und wie deren Nutzen und Aufgabenangemessenheit beurteilt wird.

Die Resultate der Untersuchung zeigten, dass ein grosser Teil der befragten Benutzer die Potentiale des Systems nicht nutzt, sondern sich auf einfache Abfragen beschränkt und mit dem Ergebnis der Recherchen – und dem System – sehr zufrieden ist. Offensichtlich ist es für die überwiegende Mehrheit der Benutzer möglich, unter Ausnutzung eines relativ geringen, aber einfach handhabbaren Teils der Gesamtfunktionalität in effizienter Weise zu den gewünschten Ergebnissen zu gelangen.

Allerdings werden die komplexeren Abfragemöglichkeiten als eher kompliziert beurteilt. Offenbar bietet das System den Benutzern gute Unterstützung für gezieltes Suchen, ist aber für relativ unstrukturierte Recherchen für viele 'Normalbenutzer' zu kompliziert in der Bedienung. Es stellt sich die Frage, ob die Gestaltung der komplexeren Abfragen zu sehr von bibliothekarischem Denken – das heisst, der Sicht von Spezialisten – mitgeprägt wurde. Hier macht sich die Schwierigkeit der Gestaltung eines öffentlichen Systems bemerkbar, das einem heterogenen Benutzerkreis mit entsprechend unterschiedlichen Aufgaben, Unterstützungsbedürfnissen und Vorkenntnissen sowie stark variierender Benutzungshäufigkeit Rechnung tragen sollte.

Um auch den 'Normalbenutzern' den Zugang zu den komplexeren Abfragen zu erleichtern, gibt es prinzipiell zwei Möglichkeiten:

(1) Verbesserung des Ausbildungsstandes (durch erweitertes Kursangebot, Abgabe einer Detailanleitung und eines 'Quick Reference Manual') und

(2) Änderung des Systems.

Verschiedene, mit vertretbarem Aufwand kurzfristig realisierbare Änderungen können hinsichtlich Maskengestaltung, Sprache, Abkürzungen, Dia-

logsteuerung usw. erfolgen; hierzu konnte den Kommentaren der Benutzer in den Fragebogen eine Fülle von Anregungen entnommen werden.

Mittel- und längerfristig ist allerdings an eine konzeptionelle Überarbeitung des Systems unter Berücksichtigung neuer Dialogtechniken und moderner Medien (wie z.B. CD-ROM) zu denken. Als eine erste Grundlage für diese Weiterentwicklung sollten die Ergebnisse eines Workshops von Benutzern und Entwicklern dienen.

3 WORKSHOP

Generelles Thema des Workshops waren Verbesserungs- bzw. Erweiterungsmöglichkeiten des Systems. Im einzelnen wurden folgende Ziele benannt:

(1) Hinweise, Vorschläge für eine Systemverbesserung erarbeiten,

(2) gegenseitiges Verständnis von Benutzern und Entwicklern fördern (Entwickler sehen, wie Benutzer mit dem System umgehen; Benutzer sehen technische Probleme der Entwickler),

(3) Kontakte der Benutzer sowie der Entwickler untereinander fördern.

Der Workshop sollte insgesamt einen halben Tag dauern und mit einem gemeinsamen Essen abgeschlossen werden. Teilnehmer des Workshops waren – neben den Entwicklern und einigen Bibliothekaren – 24 repräsentative, qualifizierte, kritische und interessierte Benutzer, die aufgrund ihrer Angaben im Fragebogen ausgewählt und eingeladen wurden (ca. 30% der befragten Benutzer hatten sich im Fragebogen bereit erklärt, an einem Workshop teilzunehmen!). Der Ablauf des Workshops gestaltete sich wie folgt:

(1) Einleitend wurden Ziel und Zweck der Veranstaltung sowie die Umfrageergebnisse vorgestellt.

(2) Anschliessend wurde in fünf themenspezifischen Arbeitsgruppen, deren Moderation je ein Softwareentwickler zusammen mit einem Bibliothekar übernahm, über das System diskutiert, anhand von Beispielen am Bildschirm wurden on line Probleme aufgezeigt und Verbesserungsvorschläge erarbeitet.

(3) Es folgte – im Plenum – die Ergebnispräsentation der Arbeitsgruppen. Vor dem gemeinsamen Essen ergab sich dann auch eine Diskussion über Möglichkeiten einer weiteren Kooperation.

Der Workshop wurde von allen Teilnehmern als erfolgreicher Beginn einer verstärkten Zusammenarbeit gewertet; die gesetzten Ziele wurden erreicht:

(1) Es ergab sich wiederum eine ganze Reihe von Verbesserungsvorschlägen; in einer gemeinsamen Sitzung der Moderatoren der Arbeitsgruppen wurden diese Vorschläge aufgearbeitet und einige davon auch sofort umgesetzt, was von den Benutzern mit Freuden festgestellt wurde.

(2) Die Benutzer erkannten dadurch, dass der Workshop keine Alibiübung war, sondern dass ihre Beiträge ernst genommen wurden. Ausserdem erhielten die Teilnehmer eine Zusammenfassung der aufgearbeiteten Workshopergebnisse (Feedback!).

(3) Ferner wurde auch das gegenseitige Verständnis von Entwicklern und Benutzern gefördert, gab es doch auf beiden Seiten einige 'Aha'-Erlebnisse zu verzeichnen.

So ist es jetzt z.B. den Benutzern verständlich, dass bei der bestehenden Datenbank das Rückwärtsblättern in Maskenfolgen nur mit immens hohem Programmieraufwand realisierbar ist; die Entwickler haben z.T. mit Erstaunen zur Kenntnis genommen, dass die Gedankengänge der Benutzer bei Recherchen im System nicht immer mit den von ihnen realisierten Dialogabläufen übereinstimmen. Schliesslich haben die Benutzer untereinander ihre vielfältigen und z.T. widersprüchlichen Anforderungen kennengelernt. Alle Teilnehmer hatten das Bedürfnis nach weiterer Zusammenarbeit; als Möglichkeiten – die sich nicht ausschliessen, sondern kombinierbar sind – wurden die Einrichtung einer Mailbox, die Bildung einer User-Group, die Ernennung eines Benutzerkoordinators sowie weitere Workshops genannt.

Zur Zeit wird ein zweiter Workshop geplant; ausserdem befindet sich die Etablierung einer gemischt zusammengesetzten Gruppe im Diskussionsstadium (Softwareentwickler, Bibliothekare und weitere Benutzer). Aufgaben dieser Gruppe wären die Durchführung weiterer Aktivitäten mit Benutzern sowie – darauf aufbauend – die Erarbeitung von Konzepten für ein zukünftiges System bis zur Erstellung von Prototypen.

Dieses Fallbeispiel zeigt, wie die – gerade bei einem öffentlichen System mit grösstenteils unbekannten Benutzern – traditionellerweise vorhandene Distanz zwischen Entwicklern und Benutzern durch eine Umfrage und einen Workshop überbrückt und die weitere Entwicklung in eine kooperative Form überführt werden kann.

Fall 4: Standardsoftware bei der ADI

1 AUSGANGSLAGE

Das diesem Fallbeispiel zugrundeliegende, gemeinsam zwischen dem Institut für Arbeitspsychologie der ETH Zürich und der ADI Software, Karlsruhe, durchgeführte Forschungsprojekt BOSS (Benutzerorientierte Softwareentwicklung und Schnittstellengestaltung) hatte die Aufgabe, vorhandene Kriterien und Methoden zu sammeln, zu systematisieren, weiterzuentwickeln sowie Benutzungstests zu unterziehen. In der Zusammenarbeit von Arbeits- und Organisationspsychologen und dem Softwarehaus (ADI) wurden die arbeitswissenschaftlichen Vorgaben aufbereitet und in Realisierungsvarianten eines konkreten Produktes (Datenbanksystem ADIMENS) erprobt. Die vorgegebenen Kriterien wurden in eine Relation zu den hard- und softwaretechnischen Möglichkeiten gesetzt und unter Mitwirkung der Endanwender sukzessive in der Produktentwicklung berücksichtigt. Die erstellten Produktvarianten wurden dabei im arbeitspsychologischen Labor ausführlichen Tests hinsichtlich der Erreichung der angestrebten Ziele (z.B. benutzerorientiertes Gestaltungskonzept) unterzogen und entsprechend verbessert. Aufgrund der weiten Verbreitung des Produktes (ca. 35'000 vollgrafische Installationen im beruflichen und privaten Einsatzkontext, insgesamt ca. 85'000 Installationen von Personalcomputern bis zu UNIX Workstations im Markt) wurde auch die Möglichkeit genutzt, einen grösseren Benutzerkreis in diesen Prozess einzubeziehen. Parallel zur Produktentwicklung untersuchte das BOSS-Projekt typische Softwareentwicklungsprozesse im Hinblick auf die Benutzerpartizipation bei der ADI.

Im folgenden werden erste betriebliche Erfahrungen und eingeschlagene methodische Wege im Kontext der benutzerorientierten Standardsoftwareentwicklung des vollgrafischen Datenbanksystems ADIMENS aufgezeigt. Das Datenbanksystem ADIMENS eignet sich aus drei Gründen besonders als dynamische Basis für die Benutzerpartizipation bei der Dialoggestaltung:

(1) *Adimens* bietet *zwei* verschiedene Benutzungsoberflächen auf *einer* einzigen Anwendungskomponente: ASCII = menüorientiert mit Funktionstasten (CUI = 'character oriented user interface'), DESKTOP = desktoporientiert unter direktmanipulativer Nutzung von Maus und Ikonen (GUI = 'graphic oriented user interface').

(2) Für experimentelle Untersuchungen stehen wegen der hohen Anzahl von Installationen Anfänger, Fortgeschrittene und Experten als Testbenutzer zur Verfügung (siehe Benutzungstests im Teil B, Kapitel 4.14).

(3) Der dem *Markenartikelmarkt* vergleichbare Markt der Standardsoftware auf Personalcomputern, zu dem *Adimens* gehört, weist *Charakteristika* auf, die für *Benutzerpartizipation* geeignet sind:
• Die *Lebens-Versions-Zyklen* der Software werden einhergehend mit dem Preis-Leistungs-Rennen auf der Hardwareseite ständig kürzer.
• Softwareanbieter treffen zunehmend auf *Neueinsteiger* in der PC-Welt, die einerseits die Qualität der gekauften Ware ähnlich beurteilen wie bei Artikeln des Alltagsbedarfs, andererseits *aus der anonymen Nach-dem-Kauf-Situation herausbrechen wollen* mit dem Wunsch nach persönlicher Beratung und Kontaktaufnahme.
• Langzeitbenutzer geben neben kritischen Hinweisen, auch ohne grössere Aufforderung, konkrete Vorschläge zur Implementierung für sie wichtiger Funktionen. Andererseits ermutigt die hohe 'Update-Treue' einen Softwarehersteller, benutzerseitig eingebrachte Verbesserungen bez. Handhabung und Funktionalität als neue Versionen auf den Markt zu bringen.

Schliesslich kann die bereits beim Entstehungsprozess stattfindende Benutzer-Entwickler-Kommunikation eine kulturell notwendige Anpassung der Programme bewirken. Dies ist allerdings mit einem Aufwand verbunden, der von US-amerikanischen Unternehmen nicht geleistet werden will und somit von lokalen Unternehmen übernommen werden muss.

2 PROJEKTMANAGEMENT UND PROJEKTABLAUF

Zu Projektbeginn lag ADIMENS in seiner zweiten Generation vor (Version 2.*). Die vollgrafische DESKTOP-Variante (Version 2.1) und die ASCII-Variante (Version 2.23) wurden, basierend auf einem gemeinsamen Datenbankkern (AdiPROG), organisatorisch getrennt entwickelt. Als Partizipationsangebote dienten in eher passiver Weise Kundenkontakte im Support, aber auch bereits in der Öffentlichkeit verkaufsfördernde Veranstaltungen (Messen, Roadshows, Händlertage, Demonstrationen, Schulungen bei einzelnen Kunden). Der empirische Vergleich beider Varianten lenkte jedoch – infolge seines überraschenden Ergebnisses hinsichtlich der Lernförderlichkeit und Überlegenheit der vollgrafischen DESKTOP-Variante auch bei Experten – die Konzentration auf die Weiterentwicklung dieser DESKTOP-Variante. Die gleichzeitig eingebrachten Gestaltungshinweise und funktio-

nellen Anforderungen erforderten in der Entwicklungsorganisation der ADI datenbankseitig die Bereitstellung eines neuen Datenbankkernes sowie die grundlegende Auseinandersetzung mit den existierenden bzw. sich am Markt abzeichnenden Oberflächensystemen, um Einschränkungen der benutzten DESKTOP-Variante langfristig zu überwinden. Daneben wurden die parallelen Weiterentwicklungen synchronisiert. Gleichzeitig wurde das Partizipationsangebot aktiver und systematischer:

(1) Bei Vor-Ort-Aktivitäten (Messen, Händlertag) wurden Benutzerzufriedenheit und Gestaltungshinweise im Kundengespräch akzentuiert.

(2) In die existierende Fehlerdatenbank für die Produktentwicklung wurden für die Weiterentwicklung auch Benutzerwünsche aufgenommen.

(3) Die Sprechstundenzeit in der Kundenbetreuung (Hotline) wurde erhöht und mit Mitarbeitern aus Vertrieb, Dokumentation und Entwicklung besetzt.

(4) Ein Benutzertreffen mit ausgewählten Langzeitanwendern wurde veranstaltet.

(5) Gemeinsam mit einer Fachzeitschrift wurde eine Fragebogenaktion durchgeführt, um das Wissen über die Benutzer (Benutzerprofil) zu vertiefen.

(6) Zu ausgewählten Aufgabenstellungen wurden induktive und deduktive Benutzungstests durchgeführt; diese Benutzungstests dienten einerseits zur Überprüfung der getroffenen Gestaltungsmassnahmen und andererseits zur Gewinnung neuer Gestaltungsvorschläge (siehe Abbildung 16).

Während die Massnahmen (1) bis (4) vor allem Verbesserungswünsche bez. Funktionalität und Handhabbarkeit lieferten, zeigte (5) darüber hinaus sowohl durch die starke Resonanz (220 Rücksendungen) als auch durch die inhaltliche Detaillierung (z.T. über 100 Seiten Begleittext und -skizzen) eine hohe Partizipationsbereitschaft. Allerdings liessen gerade (4) und (5) wegen des teilweise erheblich unterschiedlichen Anwendungskontextes auch Probleme der Singularität der gewonnenen Hinweise erkennen. Mit Hilfe der Anregungen der Benutzer wurde ein Jahr nach der Vorstellung der Version 2.3 die Version 3.0 im Herbst 1989 im Markt eingeführt.

Mit ADIMENS 3.0 erhielten die Benutzer wesentliche Pluspunkte für ein aufgabenangemesseneres Arbeiten (z.B. Interaktiver Join, Arbeitsumgebung, Multiples Sorting). Die Auswirkungen der umgesetzten Gestaltungs-

vorgaben wurden in entsprechenden Benutzungstests auf ihre Gebrauchstauglichkeit überprüft.

Die positiven Erfahrungen bei der Entwicklung führten organisatorisch bei der ADI zu einer stärker synchronisierten Parallelisierung der Projektgruppen unter der 'Patenschaft' je eines Produktverantwortlichen aus Vertrieb und Entwicklung mit frühzeitiger Einbeziehung der Dokumentation und interner Benutzer, z.B. aus dem Aufgabenfeld der Kundenbetreuung. Bei der Weiterentwicklung der Produkte werden seither Beta-Tests mit externen Benutzern durchgeführt, die auch die Benutzbarkeit berücksichtigen.

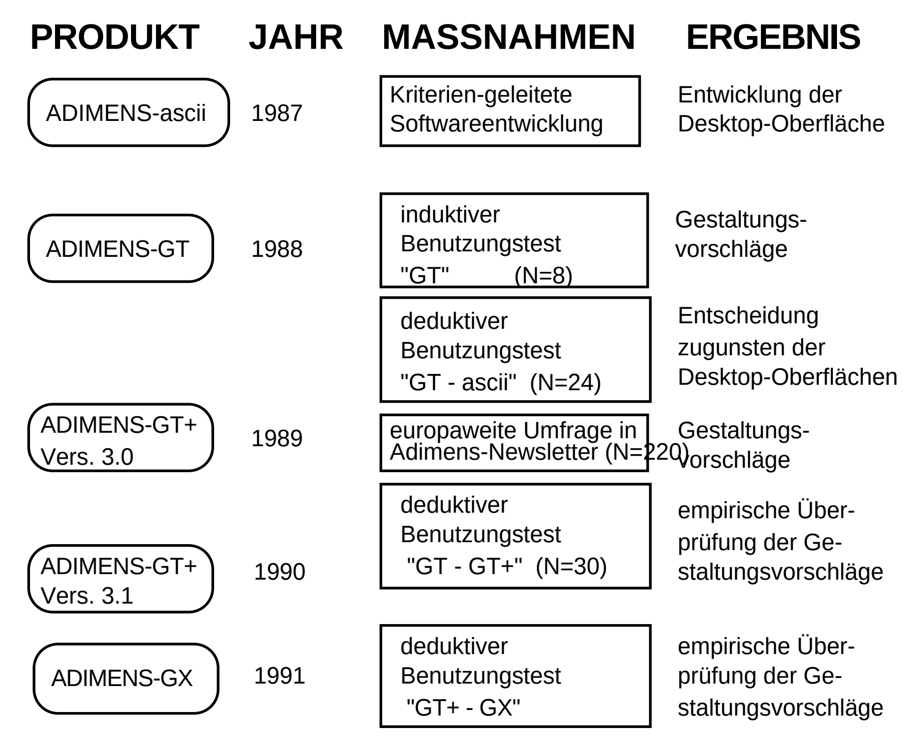


Abbildung 16: Übersicht über den Verlauf der durchgeführten Benutzungstests und Umfragen im Rahmen der Standardsoftwareentwicklung von ADIMENS in den Jahren 1988 bis 1991.

Als neues partizipatives Instrument werden seither jedem Produkt Registrierkarten beigelegt, deren Kurzfragebogen auf der Rückseite wichtige

Hinweise zum Benutzerprofil, Anwendungskontext und Raum für Kritik bietet. Die Ergebnisse der regelmässigen Auswertung dieser Informationen beeinflussten in erheblichem Mass Leistungsmerkmale und Handhabbarkeit der – wiederum nach einem Jahr – fertiggestellten Version 3.1 (Bildverwaltung, Query-by-Example, Datentyp Zeit, u.v.m.).

Parallel zur Version 3.1 entstand dann eine ADIMENS-Variante unter Windows 3.0, dessen grafische und Multitaskingmöglichkeiten die Implementierung weiterer grundlegender Gestaltungsvorgaben zulassen (Öffnen mehrerer Datenbanken, Multiwindowing auch bei Masken, direktmanipulativer Datenaustausch auch zwischen unterschiedlichen Anwendungsprogrammen). Zur Umsetzung der in der zweiten und dritten Generation benutzerseitig geforderten Leistungsmerkmale (z.B. data dictionary) wird der oben erwähnte neue Datenbankkern für die Anwendungskomponente verwendet.

3 FAZIT

Die gemachten Erfahrungen können im Hinblick auf eine benutzerorientierte Softwareentwicklung im Bereich der Standardsoftware aus der Sicht des entwickelnden Softwarehauses zu drei miteinander in Wechselwirkung stehenden Anforderungen zusammengefasst werden:

(1) In der *Aufbauorganisation des Softwarehauses* sind Massnahmen zu treffen, die über die marktnotwendige Kundenorientierung und damit verbundene verkaufsfördernde Strategien zum Absatz der entwickelten Produkte hinausgehen in bezug auf:

(a) die Förderung der Orientierung am Benutzer in allen Unternehmensbereichen durch internen exemplarischen Einsatz von Produkten (inkl. Varianten) so früh wie möglich (Beta-Test), so breit wie möglich (Marketingdatenbank, Vertriebssteuerung und Auftragsabwicklung, Produktdatenbank), verbunden mit regelmässigen internen Präsentationen bereits im frühen Entwicklungsstadium (aktuelle Auseinandersetzung mit repräsentativen Benutzern in unmittelbarer Nähe sowie Anreicherung des Pflichtenheftes mit aktuellen Verbesserungsmöglichkeiten, z.B. aus der Kundenbetreuung);

(b) die Auffächerung des Kundenkontaktes im Unternehmen sowie auf externen Veranstaltungen unter direkter Beteiligung möglichst vieler Mitarbeiter;

(c) die Verfügbarkeit einer Ablauforganisation, die Resultate aus den mittels (a) und (b) geöffneten Kanälen der Benutzerbeteiligung in neue Produkte oder Dienstleistungen (Training, Workshops) einfliessen lässt.

(2) Benutzerorientierte Softwareentwicklung benötigt für ihre konsequente Umsetzung ein Qualitätssicherungssystem, das:

(a) die Produktqualität auf Arbeitsqualität zurückführt und jeden Mitarbeiter durch seine verantwortungsbewusste Arbeit an der Qualität derjenigen Produkte und ihrer Komponenten beteiligt, die sein Aufgabengebiet berühren;

(b) im gesamten Entstehungsprozess an den Stationen eines Qualitätskreises (nach DIN ISO 9004) Möglichkeiten der Benutzerpartizipation berücksichtigt;

(c) durch hinreichende Systematisierung die fortlaufende Verbesserung der Produkte mit dem Ziel einer menschengerechten Gestaltung von Software fördert.

(3) Als wesentliche Instrumente der hier erprobten Softwareentwicklungsmethode, die eine benutzerorientierte Softwareentwicklung im Detail gewährleisten, sind zu nennen:

(a) Benutzerprofil und -kontakte (von der Breite bis in die Tiefe): Registrierkarte mit Kurzfragebogen zum Benutzerprofil, Anwendungskontext und Raum für Kritik, Berichterstattung in Fachzeitschriften zur Aktivierung der Leser, Anwender, Vor-Ort-Aktivitäten (Messen, Handel), Telefonnotizen (Hotline) und Wunschzettelbriefe in der Kundenbetreuung, Diskussionen mit Benutzern (z.B. für die Lastenhefterstellung, telefonisch oder über Benutzertreffen);

(b) Mitarbeiterbeteiligung: produktverantwortliche 'Paten' aus Vertrieb und Entwicklung auch noch nach begonnener Vermarktung, Kundenbetreuung in personeller Mischbesetzung aus den Bereichen Vertrieb, Dokumentation, Entwicklung, Projektdurchführung parallelisiert in Gruppen aus den Bereichen Marketing, Vertrieb, Dokumentation, Entwicklung von Anfang an, Mitarbeiter des Unternehmens als repräsentative Benutzer;

(c) Produktqualität: Fehler-, Wunschdatenbank und Versionshistorie innerhalb eines Qualitätssicherungssystems zur Unterstützung bei

der Produktfestlegung und -entwicklung, Qualitätsprüfung sowie der Kundenbetreuung.

Die entstandenen Strukturen und Instrumente können neben Bewertungs- und Gestaltungsschemata ein Softwareengineering für eine menschengerechte Gestaltung von Software unterstützen. Die gemachten Erfahrungen zeigen deutlich, dass Innovationen unter einer ganzheitlichen Sichtweise erforderlich sind. Einer Sichtweise, die den Benutzer als autonomes, sich qualifizierendes Subjekt sieht. Dies steht im Einklang mit dem im Rahmen des Projektes entstandenen Integrationsmodell für die Gestaltung partizipativer Softwareentwicklungsprozesse.

Fall 5: Die Einführung von PPS

In diesem Fall werden die Ziele, das Vorgehen und die damit verbundenen Probleme bei der Einführung eines rechnergestützten Produktionsplanungs- und Steuerungssystems (PPS) in einem grösseren Unternehmen beschrieben. Das Beispiel verdeutlicht die besondere Bedeutung arbeitsorganisatorischer Aspekte bei der Einführung komplexer, integrierter Softwarelösungen.

Im zeitlichen Verlauf des Projektes sind drei wesentliche Hauptphasen zu unterscheiden: Phase I, geprägt von einer 'eher arbeitsorientierten Idee'; Phase II mit einer 'technikorientierten Realisierung'; Phase III mit der 'arbeitsorientierten Korrektur'.

1 PHASE I: DIE 'EHER ARBEITSORIENTIERTE IDEE'

Zur Reduktion von Lagerbeständen und Durchlaufzeiten sowie vor allem auch zur Erhöhung der Termintreue sollte im Rahmen eines Projektes in einem grösseren Unternehmen das alte, batchorientierte PPS-System durch ein neues, integriertes und interaktives PPS-System abgelöst werden. Für die Erreichung dieser Ziele wurde u.a. vorausgesetzt, dass die Abteilungen Entwicklung, Verkauf, Beschaffung, Disposition und Arbeitsvorbereitung organisatorisch stärker integriert werden; dazu sollte eine funktional integrierte Fertigungsadministration (Disposition) geschaffen werden, welche Aufgaben der Verkaufslogistik, Entwicklungsadministration, Arbeitsvorbereitung und Beschaffung (Termin- und Mengenverantwortlichkeit für Bestellaufträge) wahrnimmt. Die informationstechnische Vernetzung dieser Organisationsstruktur sollte durch das neue System ermöglicht werden. Die organisatorischen Ziele enthielten auch die Vorstellung, die Auftragsfreigabe sowie die Terminfeinplanung (über ca. eine Woche) in die Fertigung zu verlagern. Bis zu diesem Zeitpunkt wurde die Fertigung durch das 'Terminbüro' in sehr intransparenter Weise mit Aufträgen gespeist. Im Rahmen des Projektes sollte und wurde diese 'Instanz' daher abgeschafft.

Bereits vor der Entwicklung dieser technisch-organisatorischen Ideen war im Rahmen eines neuen Informatikkonzeptes ein integriertes PPS-Standardsystem – welches u.a. Funktionen zur Kundenauftragsverwaltung, Produktionsprogrammplanung, Material- und Kapazitätsbedarfsplanung, Materialwirtschaft, Arbeitsplan- und Stücklistenverwaltung, Fertigungsauftragsverwaltung und -freigabe sowie Lagerbewirtschaftung enthält und entsprechend die gesamte logistische Kette technisch unterstützt – durch die Informatikabteilung favorisiert und von der Geschäftsleitung zur Einfüh-

rung genehmigt worden. Die genannten organisatorischen Ideen des Projekts hatte der Leiter der Abteilung Informatik vor dem Hintergrund der Möglichkeiten dieses Systems entwickelt.

2 PHASE II: DIE 'TECHNIKORIENTIERTE REALISIERUNG'

Die eher arbeitsorientierte Idee des Projektes – mit der Einführung eines technisch integrierten PPS-Systems *auch* eine funktional stärker integrierte Organisationsstruktur zu schaffen – wurde im Rahmen des Projektes zunächst nicht realisiert. Im Anschluss an die formulierten Ideen folgte kein weiterer Projektschritt, in dem verschiedene technisch-organisatorische Varianten ausgearbeitet wurden. Uneinigkeit in der Geschäftsleitung und auch unter den Abteilungsleitern bezüglich der technisch-organisatorischen Erneuerung (Pro- und Contra-Lager!) war einer der zentralen Gründe dafür, weshalb man sich zunächst auf die Einführung des technischen Systems konzentrierte.

Dazu wurde zunächst eine 'Projektgruppe' eingerichtet, die sich aus zirka 30 Mitarbeitern der verschiedenen projektrelevanten Bereiche zusammensetzte. Die Mitarbeit erfolgte ohne spezifische Qualifizierung für die Projektarbeit sowie ohne zeitliche Freistellung neben dem Tagesgeschäft. Die Projektorganisation war durch eine unklare Führungsstruktur und Kompetenzregelung gekennzeichnet. Die genannten Merkmale führten zu einer Form der Projektarbeit, die durch Schwerfälligkeit und Ineffizienz gekennzeichnet war. Diese Projektgruppe wurde daher aufgelöst und durch eine neue Projektgruppe ersetzt.

Diese neue Projektgruppe setzte sich aus drei Vertretern der internen Informatik, fünf Fachbereichsvertretern (aus der Disposition, Arbeitsvorbereitung, Qualitätssicherung und Entwicklung) sowie – zeitweise – einem Berater des Softwarelieferanten zusammen. Als Projektleiter fungierte der Leiter der Informatikabteilung (vgl. Abbildung 17). Vom Projektausschuss und Projektleiter wurden als Fachbereichsvertreter bewusst junge Mitarbeiter mit nicht allzu langer Betriebszugehörigkeit und guten fachlichen Qualifikationen ausgewählt. Sie wurden für die Mitarbeit in der Projektgruppe vom Tagesgeschäft freigestellt. Hierarchisch waren die Fachbereichsvertreter auf Gruppen- und Abteilungsleiterebene angesiedelt oder wurden nach Abschluss des Projektes in solche Positionen befördert. Diese Fachbereichsvertreter wurden spezifisch in Fragen des Projektmanagements durch eine externe Institution wie auch vom Systemlieferanten systemspezifisch ausgebildet.

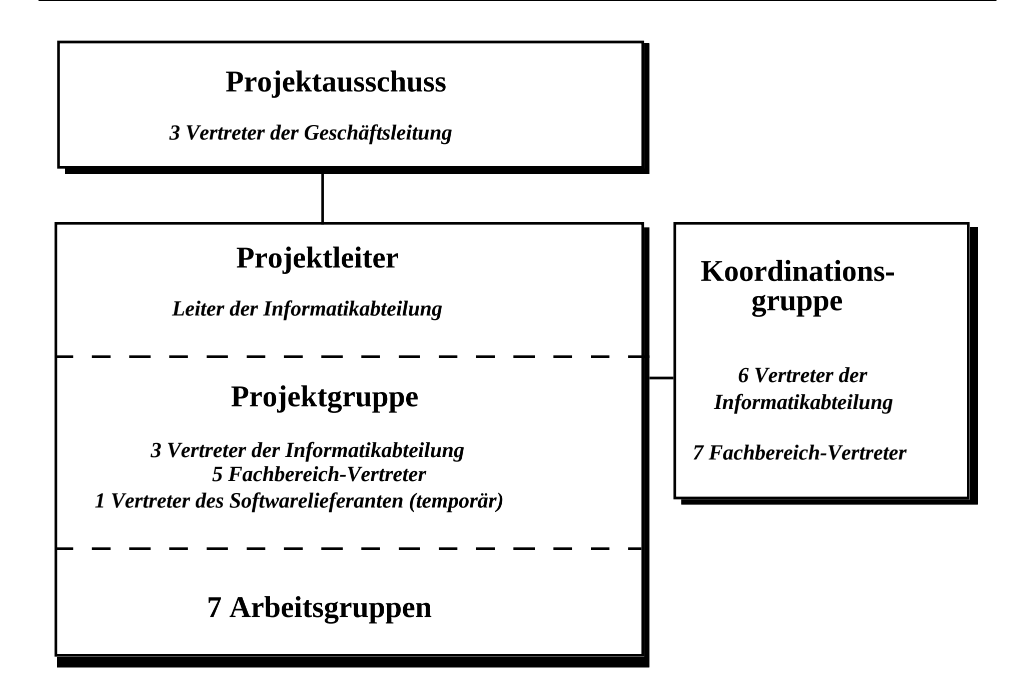


Abbildung 17: Die Aufbauorganisation des Projektes.

Die gesamte Projektgruppe wurde in bereichsspezifische Arbeitsgruppen mit je zwei bis drei Personen unterteilt. Die Verantwortung für die einzelnen Arbeitsbereiche lag jeweils bei einem Mitglied der Projektgruppe. Die einzelnen Mitglieder der Projektgruppe arbeiteten dabei in mehreren Arbeitsbereichen mit. Insgesamt sieben Arbeitsbereiche wie Bedarfsplanung, Bestandesführung, Materialklassifikation usw. wurden in diesem Sinne definiert. Zum Teil wurden auch ad hoc andere Mitarbeiter der Fachbereiche zu spezifischen Fragen beigezogen.

Die Projektgruppe wurde für die Dauer des Projektes in einem Grossraumbüro örtlich integriert, was sich als sehr kommunikationsförderlich erwies und zu einem guten Klima beitrug. Die gesamte Projektgruppe tagte wöchentlich. Die Ergebnisse der Arbeitsgruppen sowie der wöchentlichen Sitzungen wurden in Beschlussprotokollen – unterschiedlich gut und genau – dokumentiert und bei Bedarf durch den Projektleiter an den Projektausschuss weitergeleitet. Der Projektausschuss setzte sich aus drei Vertretern der Geschäftsleitung zusammen und tagte in sehr unregelmässigen Abständen (im Durchschnitt etwa alle zwei Monate).

Neben den bereits beschriebenen Gremien gab es zusätzlich eine Koordinationsgruppe, welcher ebenfalls der Projektleiter vorstand. Diese Gruppe setzte sich aus sieben Fachbereichsvertretern und sechs Vertretern der Informatikabteilung zusammen. In dieser Koordinationsgruppe wurden im Rahmen von zweiwöchigen Sitzungen Anpassungen und Schnittstellenfragen zu bereits implementierten Applikationen (wie z.B. zur Finanzbuchhaltung) geklärt.

Da die Wahl des technischen Systems ohne detaillierte innerbetriebliche Ist-Analyse und ohne weitere Konzeptspezifikation bestimmt worden war (das gewählte System teilte auch die Projektgruppe zeitweise in ein Pro- und ein Contra-Lager!), stand im weiteren Projektverlauf die Beantwortung folgender Fragen im Vordergrund: Was kann das System? Welche Situation ist in unserer Firma vorzufinden? Wie müssen wir uns verändern, damit wir systemkompatibel werden und trotzdem "unsere Organisation beibehalten können"?

Entsprechend wurden in der Folge die bestehenden Arbeitsabläufe und Informationsflüsse mittels daten- und funktionsorientierten Methoden analysiert und die 'notwendigen' systemkompatiblen Abläufe konzipiert. Die Arbeitsteilung zwischen und in den Abteilungen wurde dabei allerdings nicht mehr hinterfragt. Die systemkompatiblen Konzepte wurden in den verschiedenen Arbeitsgruppen spezifiziert. Im Rahmen dieser Arbeitsschritte wurden z.B. Stücklistenstrukturen festgelegt, Materialien zu Klassen zusammengefasst, Auftragsfreigabeverfahren definiert usw.

In dieser Situation bestand für die Projektgruppe ein grosses Problem in der Komplexität des integrierten PPS-Systems, das heisst, der vollständigen Ermittlung und Erfüllung aller systembedingten Anforderungen. Verschärfend wirkte der grosse Zeitdruck für die Realisierung dieser Projektschritte. Die programmtechnischen Systemanpassungen der Spezifikationen wurden durch Mitarbeiter der internen Informatikabteilung geleistet. Zu Simulations- und Testzwecken wurden Prototypen, die im Laufe des Projektes vom Systemlieferanten mit Echtdaten geliefert wurden sowie ein Testsystem mit der jeweils aktuellen Systemversion eingesetzt. Nach erfolgreichen Modul- und Systemtests wurden die Daten vom alten System vollständig auf das neue System überspielt. Das neue PPS-System wurde schliesslich 'über Nacht' in Betrieb genommen.

Die Mitarbeiter der verschiedenen Fachabteilungen wurden das erste Mal zirka neun Monate vor der Einführung durch die Hauszeitung über das Projekt informiert; im weiteren flossen die Informationen über 'informelle Kanäle'. Der grosse Zeitdruck während der Projektrealisierung war einer der Gründe, warum die Mitarbeiter nicht angemessen aufgeklärt, vorbereitet oder stärker in die Veränderung einbezogen wurden.

Als Folge machten sich zum Teil Desinteresse, Abneigung und auch Angst unter den Mitarbeitern gegenüber der technischen Neuerung bemerkbar; diese Problematik bekam die Projektgruppe u.a. an Bemerkungen wie "PPS wird das dann schon machen!" zu spüren.

Die Schulung der Mitarbeiter wurde ebenfalls von der Projektgruppe organisiert, dokumentiert und durchgeführt. Zunächst wurden ca. 150 Mitarbeiter geschult; die Schulung erfolgte z.T. vor, mit und nach der Einführung. Im Rahmen der Schulungseinheiten wurde Wissen über die Bedienung des Systems sowie über neue Arbeitsabläufe vermittelt.

Als grundsätzliche Probleme erwiesen sich bei der Realisierung der Ausbildung die mangelnden praktischen Erfahrungen der Projektgruppe wie auch zum Teil die Motivation und Akzeptanz der Mitarbeiter in bezug auf die Schulung. Nachher wurde auch eingesehen, dass man ein zu statisches Schulungskonzept entwickelt und realisiert hatte, das sich zuwenig an den Abläufen und den konkreten Arbeitstätigkeiten der Beschäftigten orientierte.

Der technische Teil des Projektes von der Projektidee bis zum Abschluss der Einführung des PPS-Systems wurde im budgetierten Rahmen mit einem Aufwand von zirka 20 Personenjahren und einer gesamten Dauer von ca. 2,5 Jahren realisiert. Für die eigentliche Projektrealisierung – das heisst, die konkrete Systemspezifikation, programmiertechnische Anpassung, Schulung und Einführung – stand jedoch ein viel geringerer Zeitraum zur Verfügung. Unter sehr starkem Termindruck wurden diese Projektschritte in ca. neun Monaten abgewickelt.

Die bisherigen Projektkosten belaufen sich auf zehn Millionen Franken. Der Hauptanteil fällt dabei auf Personalkosten für freigestellte Projektmitarbeiter der Fachbereiche und der Informatikabteilung sowie auf Kosten für externe Berater und externe Schulung. Die Kosten für die interne Schulung machen zirka 20%, Hardware- und Softwarekosten ebenfalls zirka 20% der bisher entstandenen Projektkosten aus.

Insgesamt sind inzwischen etwa 250 Mitarbeiter aus allen Firmenbereichen bei ihrer Arbeit vom neuen System direkt betroffen. Das System läuft auf einem Grossrechner, an dem insgesamt zirka 240 Terminals angeschlossen sind.

3 DIE SITUATION NACH DER SYSTEMEINFÜHRUNG

Im Rahmen der Projektrealisierung stand – wie bereits erwähnt – die Einführung des technischen Systems unter weitgehender Beibehaltung der or-

ganisatorischen Strukturen im Vordergrund. Die Form der Arbeitsteilung sowie die betrieblichen Abläufe wurden also mit dem neuen System kaum verändert, sondern lediglich stärker formalisiert und – vereinzelt – stärker automatisiert. Dies zeigt sich u.a. an der veränderten Arbeitssituation der Disponenten; genauere Lagerdisponierungen, On-line-Verfügbarkeitsprüfungen sowie präzisere Terminierungen sind nun zwar möglich, dies bedeutet jedoch auch, dass die Auftragsauslösung formalisierter und zeitaufwendiger wurde.

Die Ergebnisse einer schriftlichen Befragung unter Mitarbeitern aus produktionsvorgelagerten Bereichen zeigen, dass die Arbeitssituation nach der Systemeinführung als höher beanspruchend erlebt wird. So haben die Schwierigkeit der Arbeit, das Arbeitstempo, der Leistungsdruck, die nötige Konzentration und Genauigkeit sowie Belastung und Ermüdung aus der Sicht dieser Beschäftigten zugenommen.

Am PPS-System werden vor allem zu lange Antwortzeiten kritisiert. Die aufgabenbezogene Unterstützung des Systems sowie die Nützlichkeit der Hilfsinformationen werden mehrheitlich positiv bewertet, die Einschätzung der Systemflexibilität sowie -konsistenz ist sehr unterschiedlich. Die Ausbildung für die neue Arbeitssituation wird von den befragten Systembenutzern mehrheitlich als genügend bewertet.

Kommentare der befragten Mitarbeiter zur Veränderung haben mehrheitlich einen negativen 'Unterton'. Dies zeigt sich z.B. an folgenden Äusserungen: Seit der Einführung des PPS-Systems in unserer Firma "... läuft vieles etwas zähflüssiger als zuvor ...", "... hat sich die Ablauforganisation eher negativ verändert ... ", "... sind die Spannungen im zwischenmenschlichen Bereich gestiegen...", "... funktioniert ohne Computer nichts mehr", "… ist meine Motivation gesunken".

Im Rahmen von Interviews mit einzelnen Mitarbeitern wird sehr deutlich, dass neben positiven Erlebnissen mit und nach dem Projekt – z.B. das kooperative Verhalten in der neuen Arbeitssituation – zum Teil auch Unsicherheit und Angst bei den Mitarbeitern bestanden. Exemplarisch zeigt sich dies an Äusserungen wie: "Ich war drei Monate ohne Schlaf", oder: "Dies will ich nicht mehr erleben."

In bezug auf die betriebswirtschaftliche Situation bzw. die Zielerreichung nach der Systemeinführung hatten zunächst fehlerhafte Systemeinstellungen sowie Fehler im System selbst zur Folge, dass keine Aufträge aus dem System generiert werden konnten. In dieser als äussert hektisch und spannungsgeladen erlebten Zeit brach "die Auslastung der Werkstatt ziemlich zusammen". Nachdem diese Situation bereinigt war, führten ein 'denklogi-

scher Fehler' sowie unrealistische Planzahlen im System dazu, dass erheblich zuviel disponiert und gefertigt wurde.

Klagen des Lagerpersonals bei den Meistern über 'ein platzendes Lager' wurden an die Disponenten weitergetragen und von dort aus dem Verkauf mitgeteilt. Die Kommunikationsprozesse funktionierten in dieser Situation offensichtlich nicht wie vorgesehen; vom Verkauf wurden die hohen Lagerbestände fälschlicherweise mit saisonalen Schwankungen begründet.

Diese Fehldisposition hatte für die Kapazitätsplanung enorme Konsequenzen, da die 'fehlenden' Kapazitäten zirka sechs Monate nach der Einführung mit Überstunden sowie der Einstellung von Teilzeitmitarbeitern kompensiert wurden. Nachdem dieser Fehler schliesslich erkannt worden war, fehlte es an Aufträgen und die bestehende Überkapazität wurde durch Entlassung von Teilzeitmitarbeitern sowie Arbeitszeitreduktionen wieder abgebaut.

Ca. sechs Monate später zeigte sich erneut, dass beim Aufbau eines neuen Sortiments der Kapazitätsbedarf nicht eingeplant worden war und abgebaute Kapazitäten nun eigentlich wieder notwendig wären.

Bezogen auf die betriebswirtschaftlichen Projektziele musste ca. 12 Monate nach der Systemeinführung folgendes Fazit gezogen werden:

(1) Die Lagerbestände waren – wie dargestellt – erheblich gestiegen;

(2) die Termintreue hatte sich auf Kosten gestiegener Lagerbestände verbessert;

(3) die Durchlaufzeiten blieben grösstenteils unverändert.

Im Laufe des Jahres hatte sich zudem gezeigt, dass die System-'Performanz' nicht mehr ausreichte und die Antwortzeiten erheblich zu lang waren. System-'Tuning' zur Verbesserung der Performanz führte zu mehreren Systemabstürzen mit der Konsequenz, dass die Auftragsabwicklung in der Firma fast völlig zum Erliegen kam bzw. 'nichts mehr ging'.

4 PHASE III: DIE 'EHER ARBEITSORIENTIERTE KORREKTUR'

Einige systembezogene Änderungsmassnahmen sind zirka sechs Monate später erfolgt. Das System läuft nun mit besserer Performanz und Stabilität. Inzwischen wurde auch eine Analyse der bestehenden Abläufe und der technisch-organisatorischen Bedingungen in der Firma durchgeführt. Die Ergebnisse dieser Analyse brachten erwartungsgemäss eine Reihe von Problemen des betrieblichen Alltages zum Vorschein. So wurde deutlich, dass

in der Firma z.T. sehr ineffiziente Arbeitsstrukturen und Abläufe (Parallelität, Redundanz, zu viele Schnittstellen) praktiziert werden, die durch das neue PPS-System noch zementiert wurden. Einseitiges Abteilungsdenken bzw. mangelndes Verständnis einer gemeinsamen Aufgabe, schlechte Datenpflege – und somit schlechte Datenqualität – sowie unzureichende Nutzung der Systemfunktionalität sind einige Folgen dieser Situation.

Die dargestellten betriebswirtschaftlichen Ergebnisse nach der Systemeinführung wie auch die Ergebnisse dieser Analyse hatten in der Projektgruppe wie auch in einer inzwischen neu zusammengesetzten Geschäftsleitung die Einsicht verstärkt, dass die ursprünglich angestrebten betriebswirtschaftlichen Ziele mit dieser technisch-organisatorischen Lösung allein nicht erreicht werden können; das heisst, dass der Einsatz eines technisch zwar leistungsfähigen, integrierten PPS-Systems *ohne* Veränderung der bestehenden Organisationsstrukturen die Durchlaufzeiten nicht reduziert und die Termintreue nicht oder allenfalls auf Kosten hoher Lagerbestände erhöht. Im Zusammenhang mit diesen Überlegungen wurden u.a. die ursprünglich formulierten organisatorischen Ideen wieder aufgegriffen und spezifiziert. Einige der angestrebten Veränderungen sind bereits realisiert; andere befinden sich in der Realisierungsphase. Diese Veränderungsmassnahmen weisen in eine arbeitsorientierte Richtung und sind u.a. durch Merkmale wie Teile- und Produktsegmentierung, funktionale Integration sowie erweiterte Aufgabenspektren für die Beschäftigten gekennzeichnet.

5 FAZIT

Das beschriebene Fallbeispiel zeigt die Bedeutsamkeit arbeitsorganisatorischer Aspekte bei der Einführung komplexer Softwaresysteme auf. Im Rahmen des beschriebenen Projektes wurde u.a. versäumt, ein auf den Ergebnissen einer Ist- und Schwachstellen-Analyse basierendes Produktionskonzept unter Beteiligung der betroffenen Fachbereiche, der Informatikabteilung und der Geschäftsleitung zu entwickeln. Die fehlende Einigkeit in der Geschäftsleitung sowie unter den Abteilungsleitern über die geplante technisch-organisatorische Erneuerung war sicherlich einer der zentralen Gründe dafür, warum die Realisierung organisatorischer Innovationen mit den beschriebenen Problemen erst nach der Einführung des technischen Systems erfolgte.

Bei der Gestaltung der Projektorganisation wäre zudem eine umfassendere und repräsentativere Fachbereichsvertretung notwendig gewesen. Einigen der dargestellten Probleme hätte u.a. durch eine Integration von Vertretern anderer Bereiche wie z.B. aus dem Verkauf, der Beschaffung und der Fertigung in die Projektgruppe entgegengewirkt werden können. Das Durchführen von Informationsveranstaltungen, Workshops mit einer möglichst

grossen Anzahl von Projektbetroffenen zum Inhalt und Stand des Projektes schliessen sich als weitere Forderungen daran an. Vor einem soziotechnischen Hintergrund wäre für das beschriebene Vorhaben folgender, grob beschriebener, arbeitsorientierter Ablauf begründet gewesen:

(1) Ist- und Schwachstellen-Analyse der soziotechnischen Systeme,

(2) Entwicklung eines arbeitsorganisatorischen Soll-Konzeptes; Ableitung von Anforderungen bez. Technik und Mitarbeiterqualifikation (Produktionskonzept),

(3) Realisierung des Organisationskonzeptes – z.B. im Rahmen eines Pilotes in einer Abteilung – in 'konventioneller' Technikumgebung; Durchführung der notwendigen Qualifizierungsmassnahmen,

(4) Evaluation und Optimierung der neuen Arbeitsstrukturen,

(5) Entwicklung eines Feinkonzeptes für die Hard- und die Software,

(6) Auswahl, Anpassung und Implementierung der Hard- und der Software.

Ob ein solcher arbeitsorientierter Projektablauf dabei auch zur gleichen technischen Lösung geführt hätte, bleibt in diesem Zusammenhang eine offene Frage. Neben den dargestellten Versäumnissen und Problemen war das Projekt jedoch durch eine positive 'Wendemarke' gekennzeichnet: die Phase III mit der 'eher arbeitsorientierten Korrektur'. Die Erkenntnisse, die zu diesem 'Richtungswechsel' führten, wie auch die damit verbundenen Massnahmen eröffneten die Möglichkeit, dass technologische Innovationen auch mit organisatorischen Innovationen verbunden werden und die mit dem Projekt verbundenen betriebswirtschaftlichen Ziele erreicht werden können. In der Firma besteht nun die Einsicht, dass eine Vielzahl der bestehenden Probleme nur mit neuen Formen der Arbeits- und Organisationsgestaltung gelöst werden können. Dieser 'Richtungswechsel' wurde jedoch u.a. erst durch die Neubesetzung von Leitungsfunktionen ermöglicht.

ANHANG

Checkliste 1:

Prinzipien und Hinweise für Veränderungsprozesse und die Beteiligung von Benutzern bei der Softwareentwicklung

Voraussetzung

Für einen erfolgreichen und effizienten Entwicklungsprozess müssen die (infra-)strukturellen Bedingungen in der EDV-, Entwicklungs-Abteilung stimmen; stark arbeitsteilige (Projektleiter, Analytiker, Programmierer, Codierer), funktionsbezogene (Analyse, Programmierung, Test, Wartung, Methodik) Strukturen, eine – bezüglich Werkzeugen und deren Integration – ungenügende Entwicklungsumgebung und bürokratische Vorschriften für die Projektabwicklung ('Formularkrieg' bei Anträgen, Genehmigungen, Berichterstattung usw.) verhindern ein kreatives, innovatives Vorgehen und erschweren die Beteiligung von Benutzern. *Auch die Entwickler müssen gute Arbeitsbedingungen haben, damit sie im Projekt Engagement zeigen und effizient arbeiten können!*

Prinzip 1

Ausgehend vom Soll-Konzept der Arbeitsorganisation sind die technischen Unterstützungsbedürfnisse abzuleiten, die Funktionsteilung zwischen Mensch und Computer (und damit die Systemfunktionalität!) zu konzipieren und anschliessend die Benutzungsoberfläche zu entwerfen.

Prinzip 2

Statt länger dauernde Grossprojekte besser kürzere Teilprojekte mit bald sichtbarem Erfolg für die Beteiligten realisieren! Grossprojekte in überschaubare Einheiten aufteilen! Die Motivation der Beteiligten nimmt mit der Zeit ab, wenn kein Resultat der Arbeit ersichtlich wird.

Prinzip 3

Zur Vermeidung oder Überwindung von Ungewissheiten, Ängsten und Widerständen sollten die von der Veränderung betroffenen Mitarbeiter frühzeitig und fortlaufend informiert, in den Veränderungsprozess einbezogen und gründlich ausgebildet werden: die Betroffenen zu Beteiligten machen!

Das Fachwissen der zukünftigen Benutzer durch deren frühzeitigen Einbezug in die Softwareentwicklung nutzen!
Der Einbezug der Benutzer darf aber nicht auf eine Pseudopartizipation hinauslaufen, denn sie wird meistens durchschaut und führt zu keinen längerfristig befriedigenden Ergebnissen.

Prinzip 4

Bei der Organisation der Benutzerbeteiligung ist unbedingt auf eine *repräsentative Vertretung* verschiedener Fachabteilungen, Hierarchieebenen und Benutzergruppen zu achten!

Prinzip 5

Die am Projekt mitwirkenden *Benutzer* müssen zu einem Teil ihrer Arbeitszeit von ihren täglichen Aufgaben *freigestellt* werden, um Überlastungen zu vermeiden und ihnen eine engagierte Mitarbeit zu ermöglichen. Projektaufgaben dürfen ihnen nicht als zusätzliche Arbeit aufgebürdet werden, da dies in der Konsequenz allmählich dazu führt, dass die Projektarbeit als lästige Pflichtübung betrachtet und vernachlässigt wird!

Prinzip 6

Die *Erwartungen* aller Beteiligten in bezug auf die Ziele des Projektes müssen auf einem *realistischen Niveau* gehalten werden; es dürfen keine überzogenen Versprechungen gegeben werden, die dann unter Umständen nicht eingehalten werden können! Enttäuschte Erwartungen führen zu Frustration, die sich in Passivität und Widerstand äussern und die Zusammenarbeit gefährden kann.

Prinzip 7

Es kann gefährlich sein, Projektmanagement-'Rezepte' ungeprüft zu übernehmen; besser ist es, eine dem Betrieb, dem Vorhaben sowie den personellen und infrastrukturellen Voraussetzungen *angepasste Projektorganisation* zu entwickeln!

Prinzip 8

Benutzer und Entwickler sind durch eine *komplementäre Ausbildung* auf die Zusammenarbeit vorzubereiten; die Benutzer müssen über Möglichkeiten und Grenzen des technischen Systems orientiert werden, die Entwickler

müssen die Aufgaben und Arbeitsabläufe der Benutzer gut kennen! Dies erleichtert auch das gegenseitige Verständnis und die Kommunikation!

Prinzip 9

Für *Analyse und Anforderungsermittlung* ist *mehr Zeit* als üblich einzukalkulieren! Auf *neue Wünsche und Änderungsvorschläge* der Benutzer im Laufe des Projektes muss man *gefasst sein* sowie offen und flexibel darauf reagieren!
Benutzerbeiträge nicht a priori mit technischen Argumenten abweisen ("sowieso nicht machbar"), sondern ernsthaft prüfen und allenfalls Realisierungsprobleme erläutern, denn: mehrfach abgewiesene Vorschläge der Benutzer führen zu Passivität und Resignation ("... es hat ja doch keinen Sinn...")! Benutzerbeteiligung darf keine Alibiübung sein.

Prinzip 10

Anstelle schwer verständlicher Pflichtenhefte, Systembeschreibungen usw. *Prototypen erstellen*, den Benutzern erläutern, ihnen Zeit für eine eingehendere Beschäftigung mit dem Prototyp geben und dann eine Besprechung oder einen Workshop durchführen! 'Ein Bild sagt mehr als tausend Worte'!

Prinzip 11

Nicht ausgetragene, durch Gespräche bewältigte *Konflikte* in Projekt- oder Arbeitsgruppen können das gesamte Projekt gefährden. Ebenso können andauernde Kommunikationsprobleme zwischen Benutzern und Entwicklern eine effiziente, fruchtbare und persönlich befriedigende Zusammenarbeit verunmöglichen. In diesen Fällen kann ein begleitendes *Training sozialer Fertigkeiten und der Konfliktbewältigung* hilfreich sein!

Projektorganisation

(1) Sind *alle relevanten Interessengruppen* in der Projektorganisation vertreten?
- [ ] Management
- [ ] Spezialisten (z.B. einer Stabsabteilung)
- [ ] betroffene Fachabteilungen
- [ ] externe Berater
- [ ] Betriebsrat, Personalkommission
- [ ] weitere: ....

(2) Ist die Organisation der *Benutzerbeteiligung* (bezüglich Form, Grad, Inhalt, Methoden, Zeitpunkt) *sichergestellt*?
(3) Sind Funktion, Aufgaben, Kompetenzen und Verantwortlichkeiten der gebildeten Gremien und deren Zusammenwirken – vor allem bezüglich Entscheidungsprozessen – definiert? Gibt es Unklarheiten bezüglich Zuständigkeiten, gibt es Doppelspurigkeiten oder Lücken?
(4) Sind Projekt- und Arbeitsgruppen ihren Aufgaben entsprechend personell richtig besetzt?
(5) Sind Inhalt und Zielsetzungen des Projektes definiert? Sind sie allen Beteiligten bekannt und besteht ein Konsens darüber?
(6) Besteht ein *Konzept über die Projektabwicklung* mit Zeitplan, Ressourcen, Aktivitäten usw.?
(7) Sind genügend *Zeitreserven* eingeplant? Ist speziell für Analyse, Anforderungsermittlung und Grobkonzept ausreichend Zeit vorgesehen, so dass auch mehrere Lösungsvarianten entwickelt werden können?
(8) Ist ein *Informationsverteilungsnetz* vorhanden? Sind Kommunikationskanäle geschaffen und auch allen Beteiligten bekannt?
(9) Ist ein *Dokumentationssystem* eingerichtet?
(10) Sind alle Voraussetzungen methodischer und technischer Art für den Projektbeginn überprüft und erfüllt?
(11) Sind alle Beteiligten auf dem *erforderlichen Ausbildungsstand*?

Information

(1) Durch frühzeitig beginnende, fortlaufende Information wird der Entstehung von Ungewissheiten und Gerüchten vorgebeugt und Transparenz geschaffen.
(2) Zur *Verbreitung der Information und zur Ermöglichung von Rückmeldungen* sind Kommunikationskanäle und ein Informationsverteilungsnetz zu schaffen. Möglichkeiten:
- periodische Information in der 'Hauszeitung',
- periodische, projektspezifische Mitteilungen, z.B. als Bulletins,
- Mailbox, Anschlagbrett,
- persönliches Anschreiben, Gespräch,
- Abteilungs-, Betriebsversammlung,
- Vertiefung der Information in Seminaren,
- Diashows, Filme und Modelle zur Veranschaulichung.

(3) Nicht nur die vom Projekt direkt Betroffenen, sondern auch die indirekt Betroffenen sollen über die geplanten Neuerungen angemessen informiert werden.

(4) Das *Schwergewicht* der Information soll nicht auf der detaillierten Darstellung der technischen Veränderung liegen, sondern vor allem deren Bedeutung für die *Veränderungen der Arbeitsorganisation und der Arbeitsrolle der Mitarbeiter* darlegen.
(5) Die *Erstinformation* soll den Mitarbeitern global Auskunft geben über
- die Gründe, die zum Projekt geführt haben,
- die möglichen Auswirkungen des Projektes in personeller, arbeitsorganisatorischer und technischer Hinsicht,
- das weitere Vorgehen und Möglichkeiten der Mitwirkung.

(6) Die *fortlaufende Information* soll die Mitarbeiter orientieren über
- den Stand des Veränderungsprozesses,
- entwickelte Konzepte,
- getroffene Entscheide und daraus abgeleitete, bevorstehende Aktivitäten.

Partizipation

(1) Der Einbezug der Benutzer sollte schon in der Vorbereitungs- und Planungsphase erfolgen, damit ihre Bedürfnisse berücksichtigt und ihr Fachwissen genutzt werden kann. Dies reduziert auch die Wahrscheinlichkeit, dass nachträgliche Korrekturen notwendig werden.
(2) Wer sind die zukünftigen Benutzer, was sind ihre Aufgaben? Sind die Benutzervertreter repräsentativ für die gesamte zukünftige Benutzerpopulation?
(3) Werden die Benutzervertreter für die Projektarbeit freigestellt, das heisst, von ihren sonstigen Aufgaben zum Teil entlastet?
(4) Zur *Beteiligung* der Benutzer gibt es *verschiedene Möglichkeiten*:
- mündliche und schriftliche Umfragen,
- die Mitarbeit in Projektgruppen,
- die vollständige Übertragung von Teilaufgaben an Mitarbeiter in Arbeitsgruppen,
- den Einsatz von Kontaktpersonen,
- periodisch durchgeführte Workshops usw.

Ausbildung

(1) Eine *frühzeitige* Ausbildung von *Benutzern und Entwicklern* ist die Grundlage für eine aktive, kompetente Mitwirkung und effiziente Arbeit im Projekt. Inhalte können sein:
- Organisation,
- soziale Fertigkeiten,
- Methoden, Fachwissen,

- EDV, Technik,
- Ergonomie,
- Präsentationstechniken.

(2) Eine gründliche Schulung der operationellen Systembenutzung vermindert die Befürchtungen der Mitarbeiter, die zukünftigen Anforderungen mit den bisherigen Kenntnissen nicht mehr bewältigen zu können.
Zusätzlich sollte den Benutzern ein Grundlagenwissen vermittelt werden, das ihnen zu einem besseren Verständnis der prinzipiellen Funktionsweise des technischen Systems verhilft. Die Benutzer werden sich dadurch der Technik gegenüber weniger hilflos und ausgeliefert fühlen.

(3) Die Schulung sollte den *Vorkenntnissen der Teilnehmer angepasst* sein und auf die *konkreten Aufgaben und Arbeitsabläufe bezug nehmen.*

(4) Die Ausbildungsinhalte sollten möglichst *direkt*, das heisst ohne längere zeitliche Verzögerung, in die Praxis *umsetzbar* sein.

(5) Die Einrichtung von Übungsarbeitsplätzen vor der Systemeinführung ermöglicht eine spielerische (und damit angstfreie) Angewöhnung an das zukünftige technische System.

(6) Ein verständlich geschriebenes, gut aufgebautes, nicht zu umfangreiches *Benutzerhandbuch mit Beispielen*, Erklärungen von Besonderheiten und Hinweisen auf spezielle Arbeitshilfen kann die Systembenutzung wesentlich erleichtern.
Zusätzlich empfehlenswert ist die Abgabe eines Faltblattes (im Sinne eines 'Quick Reference Guides') mit Kurzbeschreibungen der wichtigsten Befehle, der Funktionstastenbelegung, 'short cuts', häufig gebrauchten Code keys usw.

(7) Die Ausbildung sollte auch eine *Aufklärung über Möglichkeiten der individuellen Anpassung* des Computerarbeitsplatzes (Verstellbarkeit von Bildschirm, Stuhl, Tisch usw.), des ermüdungsarmen Arbeitens (Vermeidung von Blendung, Zwangshaltungen usw.), Einstellung von Farben, Kontrast, Helligkeit usw. enthalten.

Einführung

(1) Gerade bei grösseren Projekten sollte die Einführung des Systems schrittweise (sinnvoll abgegrenzte (Teil-)Anwendungen, Module) erfolgen. Die Veränderung wird dadurch für die Mitarbeiter leichter überblickbar.

(2) Jeder Teilschritt sollte für die Mitarbeiter mit *Erfolgserlebnissen* verbunden sein und Vorteile erkennbar werden lassen.

(3) Die Veränderung verlangt von allen Beteiligten einen stetigen Lernprozess; ist deshalb eine Evaluation des Systems nach der Einführung –

das heisst, nachdem es einige Zeit im produktiven Einsatz stand – vorbereitet, um eventuell noch vorhandene Mängel erkennen (und korrigieren!) zu können?

(4) Während und auch einige Zeit nach der Einführung müssen die Benutzer die Möglichkeit haben, bei Benutzungsproblemen *menschliche Unterstützung* (z.B. über eine Hotline) zu erhalten; ist dafür genügend personelle Kapazität vorgesehen und ist den Benutzern die Anlaufstelle auch bekannt? On-line-Tutorials, Hilfefunktionen und Handbücher bieten dafür *keinen Ersatz!*

Prototyping

(1) Sind *Werkzeuge* ('tools') für die Erstellung von Prototypen *vorhanden?*

(2) Herrscht bei allen Beteiligten *Klarheit über die Fragen*, die mit dem Prototyp beantwortet werden sollen?

(3) Gibt es ein *Konzept für die Austestung* von Prototypen durch Benutzer?

(4) Sind *realistische Aufgaben* (mit Testdaten) formuliert?

(5) Wird der Test durch Mitarbeiter durchgeführt, die *repräsentativ* sind für die zukünftigen Benutzer?

(6) Ist eine *schnelle Änderung von Prototypen* gewährleistet? (Eine allzu frühe Fixierung auf eine Lösung verhindert die Entwicklung alternativer, eventuell besserer Varianten!)

Checkliste 2:

VEREINBARUNGEN FÜR EINEN PARTNERORIENTIERTEN DEMOKRATISCHEN GRUPPENPROZESS

Für die erfolgreiche Arbeit in einer Gruppe können die folgenden Aspekte als wichtigste Voraussetzungen gelten:

Vereinbarung auf	Vermeidung von
+ Persönliches Engagement jedes Teilnehmers: d.h., Einigkeit über die Zielsetzung(en).	– Vorgefassten Meinungen, funktionellen Fixationen: d.h., kein Festhalten an persönlichen einseitigen Erfahrungen.
+ Jeder soll reden können, jeder soll aktiv sein: d.h., niemandem die Meinung aufdrängen (–> ausdiskutieren); nicht vorzeitig nachgeben, nur um Einigkeit zu erreichen (kein 'Kuhhandel'); 'Störungen' anmelden.	– Ungünstigen äusseren Bedingungen: d.h., kein Zeitdruck; keine Störungen von aussen; Gruppe nicht zu gross; alle Teilnehmer sind anwesend.
+ Jeder ist Experte auf seinem Gebiet, jeder ist gleichberechtigt: d.h., abweichende Meinungen sind nicht störend, sondern ein konstruktiver Beitrag.	– Autoritätsdenken, Festhalten an eingefahrenen Rollen: d.h., der Leiter drängt sich nicht in den Vordergrund; der Moderator achtet auf Demokratie.
+ Alle Hinweise aufgreifen, nichts untergehen lassen: d.h., Trennung von 'Probleme sammeln, Ideen finden' und 'Probleme diskutieren, Ideen bewerten'; alles für jeden sichtbar registrieren.	– Abschottung gegenüber neuen Ideen: d.h., keine Killerphrasen (verbal und nonverbal), wie z.B. "... ist technisch nicht machbar", "... das haben wir noch nie so gemacht".
+ Das Ergebnis der Gruppenarbeit ist Eigentum der Gruppe: d.h., kein individuelles Urheberrecht.	

Übersicht 1:

ÜBERSICHT ÜBER INDIKATOREN FÜR DAS SCHEITERN VON SOFTWAREPROJEKTEN

Nach Bartsch-Spörl (1991, S. 304ff) gibt es die folgenden typischen Situationen, die als Indikatoren für 'Projektkrankheiten' gelten können:

1. *Alle sind sich einig, dass das Projekt gemacht werden sollte, aber keiner will die Verantwortung dafür übernehmen.*
   Eine solche Situation deutet darauf hin, dass eventuell a) das Risiko des Scheiterns insgesamt noch zu hoch ist und/oder dass b) die Übernahme der Verantwortung nicht attraktiv ist, weil sie z.B. ein Übermass an Belastung mit Pflichten ohne ausreichende Rechte bedeuten würde. Im Fall A ist die Zeit für das Projekt noch nicht reif; im Fall B müssen die Rahmenbedingungen geändert werden.

2. *Die Geschäftsleitung hat ein Projekt 'angeordnet', das die Anwender nicht wollen.*
   Hier helfen nur Gespräche mit beiden Seiten mit dem Ziel, einen für alle Beteiligten gangbaren Weg zu einem eventuell veränderten Projektziel zu finden. Wenn dies nicht möglich ist, sollte man von der Durchführung dieses Projekts Abstand nehmen.

3. *Ein Projekt ist von den Ressourcen her zu knapp budgetiert worden.*
   Hier hilft entweder a) eine entsprechende Aufstockung der Ressourcen oder b) der Verzicht auf einen erheblichen Teil der Funktionalität. Ist Fall A nicht möglich und Fall B nicht sinnvoll, dann sollte man das Projekt verschieben.

4. *Die EDV-Leute verfolgen ein rein technikorientiertes Ziel (z.B. die Evaluation von CASE-Werkzeugen) und wollen das mit dem Budget der Anwender bezahlen.*
   Wenn das Projekt z.B. eine für die Firma insgesamt wichtige Infrastrukturmassnahme ist, dann muss sich das in einer veränderten Art der Finanzierung niederschlagen; wenn es jedoch niemandem einen ersichtlichen Nutzen bringt, dann gehört es beendet.

5. *Für die geplante Anwendung ist eine ungeeignete Entwicklungsphilosophie gewählt worden (z.B. Entwicklung eines Management-Informationssystems in COBOL ohne Datenbank und ohne flexible Abfragesprache).*
   Wenn der Anwendernutzen mit der gewählten Entwicklungsphilosophie nicht erreicht werden kann, muss die technologische Basis gewechselt

werden – dies ist zwar manchmal teuer, aber jedenfalls besser als eine unbrauchbare Anwendung.

6. *Der Zeitplan des Projektes gerät ins Wanken (z.B. aufgrund von nicht kalkulierten Personalfehlzeiten).*
In solchen Fällen sollte man sich keine Illusionen machen, dass man die Rückstände leicht wieder einholen kann; man sollte lieber gleich mit den Anwendern über einen späteren Einführungstermin oder über eine erste Version mit reduzierter Funktionalität zu sprechen beginnen.

7. *Es gibt keinen einzigen 'Anwendungsexperten' in der Projektgruppe (z.B. weil die Fachabteilung unter akutem Personalmangel leidet).*
Diese höchst bedenkliche Situation ist als untragbar, in wichtigen Projekten sogar als unverantwortlich anzusehen; hier gilt es unbedingt eine Lösung zu finden, die die Fachabteilung aktiv in das Projekt mit einbezieht.

8. *Die im Projekt verwendete Methodik stösst auf Ablehnung bei den Projektmitarbeitern.*
Dieser äusserst alarmierende Indikator kann bedeuten: Fall A – die Qualifikation für den Einsatz der Methodik fehlt; Fall B – die 'verordnete' Methodik gehört einer anderen 'Generation' an und setzt andere als von der zu entwickelnden Anwendung benötigte Prioritäten; Fall C – es wird ein Machtkampf zwischen altgedienten Softwareentwicklern und einem neuen Vorgesetzten ausgetragen. Die Lösung heisst: im Fall A die Projektmitarbeiter qualifizieren, im Fall B die Methodik wechseln und im Fall C eventuell abwarten, bis sich die Wogen geglättet haben.

9. *Der Entwurf hat sich 'festgefahren', das heisst, berechtigte Anforderungen lassen sich nicht mehr berücksichtigen.*
Dies ist eine Sackgasse, aus der das Projekt unbedingt herausgeholt werden muss, ggf. um den Preis eines zusätzlichen Iterationsschrittes für den gesamten Entwurf. Es ist in diesem Fall zunächst besonders wichtig, die Ursachen für die fehlende Flexibilität zu finden, denn dann kann man sinnvoll entscheiden, was beim nächsten Anlauf anders gemacht werden muss.

10. *Die Qualitätssicherung (QS) wird von der Projektleitung auf die leichte Schulter genommen.*
Die Folgen dieses unverantwortlichen Verhaltens werden die Organisation und die späteren Anwender 'schmerzhaft' zu spüren bekommen. An der notwendigen Qualität zu sparen ist der denkbar teuerste Fehler.

Übersicht 2:

ÜBERSICHT ÜBER HILFSMITTEL ZUR ENTWURFSUNTERSTÜTZUNG

'Dynamic Rules for User Interface Design' (DRUID) als Hyper-Card-Stapel für den Apple-Macintosh aus dem Jahre 1989 umfasst alle Gestaltungshinweise und Entwurfsrichtlinien der Sammlung von Smith und Mosier (1986). Diese hypertextbasierte Software ist ebenso wie die Papierversion der 'Guidelines for Designing User Interface Software' von Smith und Mosier zu beziehen bei:

U.S. Department of Commerce, National Technical Information Service (NTIS), Springfield, VA 22161, USA.

Die entsprechende IBM-kompatible PC-Version als Hypertextsystem ist unter dem Namen 'Navi Text SAM' bei der folgenden Adresse beziehbar:

North Lights – Software Corporation, 24A Pilgrim Drive, P.O. Box 1056, Westford, MA 01886, USA.

Für die Unterstützung der Analyse von Benutzerverhalten, welches mit einem Videorekorder aufgenommen wurde, steht das folgende Softwarewerkzeug zur Verfügung: 'DRUM – The Diagnostic Recoder for Usability Measurement', beziehbar bei:

National Physical Laboratory, DITC HCI Group, Teddington, Middlesex, TW11 0LW, United Kingdom, Telefon: +44 81 943 6097.

Als geeignete Hilfsmittel zur schnellen Entwicklung von Masken und Oberflächen können die folgenden Softwarewerkzeuge angegeben werden:

Animation Works (1991) Gold Disk Inc., P.O. Box 789, Streetsville, Mississauga, Ontario, Canada L5M 2C2.

Astound (1993) Gold Disk Inc., P.O. Box 789, Streetsville, Mississauga, Ontario, Canada L5M 2C2.

Hypercard (1992) Claris International Inc., 250 The Esplanade, Suite 202, Toronto, Ontario M5A 1JS. Telefon (416) 941 9611.

IconAuthor (1989) ADI GmbH, Hardeckstr. 5, D-75021 Karlsruhe. Telefon (721) 57 0000.

IconAuthor (1991) AimTech Europe Limited, 1 Carthusian Street, London EC1M 6EB (UK). Telefon (071) 250 1225.

MacroMind Director (1990) MacroMind, 410 Townsend, Suite 408, San Francisco, CA 94107 (USA). Telefon (415) 442 0200.

NeXT Step (1991) NeXT Inc., 900 Chesapeake Drive, Redwood City, CA 94063 (USA).

NeXT Interface Builder (1991) NeXT Inc., 900 Chesapeake Drive, Redwood City, CA 94063 (USA).

Prototyper (1990) Smethers & Barnes Inc., P.O. Box 639, Portland, Oregon 97207 (USA).

ToolBook (1989) ADI GmbH, Hardeckstr. 5, D-75021 Karlsruhe. Telefon (721) 57 0000.

ToolBook (1989) Asymetrix Corp., 110 Avenue N.E., Suite 717, Bellevue, Washington 98004 (USA).

VisualBasic (1993) Microsoft Corp., One Microsoft Way, Redmond, WA 98052-6399 (USA).

UIMX (1990) Visual Edge Software Ltd., 3870 Cote Vertu, Montreal, Quebec, Canada H4R 1V4, Telefon (514) 332 6430.

WindowsMAKER (1991) Blue Sky Software Corp., 2375 East Tropicana Avenue, Suite 320, Las Vegas, NV 89119 (USA). Telefon (702) 465 6365.

Folgende Guidelines, Leitfäden, Checklisten usw. sind im Buchhandel erhältlich:

Apple® (1990) Human Interface Guidelines: The Apple Desktop Interface. Amsterdam: Addison Wesley.

IBM® Systems Application Architecture (1991) Common User Access – Guide to User Interface Design. (Bestell-Nr.: SC34-4289-00), IBM Corporation, Department T45, P.O. Box 600 000, Cary, North Carolina 27512-9968, USA.

IBM® Systems Application Architecture (1989) Common User Access – Basic Interface Design Guide. (Bestell-Nr.: SC26-4583-0), IBM Corporation, Department T45, P.O. Box 600 000, Cary, North Carolina 27512-9968, USA.

IBM® Systems Application Architecture (1991) Common User Access – Advanced Interface Design Reference. (Bestell-Nr.: SC34-4290-00), IBM Corporation, Department T45, P.O. Box 600 000, Cary, North Carolina 27512-9968, USA.

INRA (1994) Der Interface Ratgeber: ein Informations- und Beratungssystem zur Software-Ergonomie. Zu beziehen bei: Prof. Dr. H. Wandke, Institut für Psychologie, Humboldt-Universität, Oranienburger Str. 18, D-10178 Berlin, Telefon: +49-30-2805115, Fax: +49-30-2829179.

ISO 9241/10 (1993) Fragebogen zur Beurteilung von Software auf der Grundlage der Internationalen Ergonomie-Norm ISO 9241/10. Dr. Prümper & Partner, Holzhofenstrasse 8, D-81667 München.

MUSiC (1993) Metrics for Usability Standards in Computing. Zu beziehen bei: National Physical Laboratory, DITC HCI Group, Teddington, Middlesex, TW11 0LW, United Kingdom, Telefon: +44-81-943 6097.

OSF/Motif™ (1990) User's Guide. Englewood Cliffs: Prentice Hall.

Siemens Nixdorf Informationssysteme AG (1992) Alpha-Styleguide – Richtlinien zur Gestaltung von Zeichenorientierten Benutzungsoberflächen. (Bestell-Nr.: U8556-J-Z147-1), Otto-Hahn-Ring 6, D-8000 München 83.

Siemens Nixdorf Informationssysteme AG (1992) Alpha-Styleguide-Checkliste V1.0. (Bestell-Nr.: U8557-J-Z147-1), Otto-Hahn-Ring 6, D-8000 München 83.

Siemens Nixdorf Informationssysteme AG (1992) Styleguide V1.0 – Richtlinien zur Gestaltung von Benutzungsoberflächen. (Bestell-Nr.: U6542-J-Z97-1), Otto-Hahn-Ring 6, D-8000 München 83.

Siemens Nixdorf Informationssysteme AG (1992) Styleguide-Checkliste. (Bestell-Nr.: U6615-J-Z97-1), Otto-Hahn-Ring 6, D-8000 München 83.

Smith S. & Mosier J. (1986) Guidelines for Designing User Interface Software. (MITRE Technical Report, Document No. AD-A177 198). Bedford MA (USA).

Software Checker (1992; version 2.0) An aid to the critical examination of the ergonomic properties of software. Zu beziehen bei: The Swedish Confederation of Professional Employees, P.O. Box 5252, S-102 45 Stockholm, Sweden, Telefon: +46-8-782 91 00.

SUMI (1993) Software Usability Measurement Inventory. Zu beziehen bei: Human Factors Research Group, Department of Applied Psychology, University College, Cork, Ireland, Telefon: +353-21-276 871.

Folgende Bücher zur softwareergonomischen Bildschirmgestaltung sind im Buchhandel erhältlich:

AWV Vordrucke 9 (1992) Arbeitshilfen für Anwenderfreundliche Bildschirm-Eingabemasken. Arbeitsgemeinschaft für wirtschaftliche Verwaltung e.V., Düsseldorfer Strasse 40, Postfach 5129, D-6236 Eschborn. (Bestell-Nr. 02-501), Telefon (06196) 495 388.

Baitsch C., Katz C., Spinas P. & Ulich E. (1989) Computerunterstützte Büroarbeit. Zürich: Verlag der Fachvereine.

Balzert H., Hoppe H. U., Oppermann R., Peschke H., Rohr G. & Streitz N. A. (1988; Hrsg.) Einführung in die Software-Ergonomie. (Reihe Mensch Computer Kommunikation – Grundwissen 1; Balzert, H., Hrsg.). Berlin: de Gruyter.

Brown C. M. (1989) Human-Computer Interface Design Guidelines. Norwood: Ablex.

Duell W. & Katz C. (1990) Ratgeber Bildschirmarbeit. (Schriftenreihe der Bundesanstalt für Arbeitsschutz, Forschungsanwendung FA 24). Bremerhaven: Verlag für neue Wissenschaften.

Galitz W. (1989) Handbook of Screen Format Design. P.O.Box 181, Wellesley, MA 02181 (USA).

Hoffmann T., Klose H-G. & Martin H. (1989) Handbuch zur softwareergonomischen Gestaltung von Bildschirmmasken. (VDI Fortschrittsberichte Reihe 10: Informatik/Kommunikationstechnik Nr. 103) Düsseldorf: VDI Verlag.

Lauter B. (1987) Software-Ergonomie in der Praxis. Wien: Oldenbourg.

Mayhew D. J. (1992) Principles and Guidelines in Software User Interface Design. Englewood Cliffs: Prentice Hall.

Rödiger K-H., Hampe-Neteler W. & Piepenburg U. (1991) Software-Ergonomie: Gestaltungsgrundsätze der DIN-Norm 66234 Teil 8 und ihre Umsetzung. Oberhausen: Technologie Beratungsstelle des DGB.

Spinas P., Troy N. & Ulich E. (1983) Leitfaden zur Einführung und Gestaltung von Arbeit mit Bildschirmsystemen. München: CW.

Wandmacher J. (1993) Software-Ergonomie. (Mensch-Computer-Kommunikation: Grundwissen, Band 2), Berlin New York: de Gruyter.

Ziegler J. & Ilg J. (1993; Hrsg.) Benutzergerechte Software-Gestaltung. München: Oldenbourg.

Zur Beurteilung und Einschätzung der verschiedenen Verfahren und Gestaltungsansätze sei auf die folgende Arbeit verwiesen:

Hampe-Neteler W. & Rödiger K-H. (1991) Software-Ergonomie: Verfahren der Evaluierung und Standards zur Entwicklung von Benutzungsoberflächen. (Fachbereich Mathematik/Informatik, Bericht Nr. 2/92). Bremen: Universität Bremen.

Übersicht 3:

ARBEITSANALYSEVERFAHREN

(in Anlehnung an Schüpbach 1993, S. 180–181)

Verfahren, Autor(en)	**Bewertungskriterien, Hauptdimensionen**	**Anwendungsbereich**
1. Beobachtungsinterviews: a) Tätigkeitspsychologisch/handlungstheoretisch fundierte Verfahren		
TBS-L – Tätigkeitsbewertungssystem (Langform) Hacker, Iwanova und Richter (1983)	Persönlichkeitsförderlichkeit: A: Organisatorische und technische Determinanten vollständiger Tätigkeiten B: Kooperation und Kommunikation C: Verantwortung D: Erforderliche psychische Prozesse und Repräsentationen E: Qualifikations- und Lernerfordernisse	Gestaltung industrieller Montage-, Bedien- und Überwachungstätigkeit: – Schaffung vollständiger Tätigkeitsinhalte – Schaffung von Handlungsspielräumen für die selbständige Planung und Ausführung komplexer Arbeitsabläufe
TBS-GA – Tätigkeitsbewertungssystem (geistige Arbeit) Rudolph, Schönfelder und Hacker (1987)	wie TBS-L	Wie TBS-L, aber für überwiegend geistige Arbeitstätigkeiten mit und ohne Rechnerunterstützung
SABA – Spezielle Analyse belastender Arbeitsfaktoren Richter, Heimke und Malessa (1988)	Beeinträchtigungslosigkeit: A: Unmittelbar gestaltbare Aufgabenmerkmale (Anzahl von Teiltätigkeiten, sequentielle Vollständigkeit, Bewegungsvielfalt, Zyklusdauer, Rückmeldung, Qualitäts-, Quantitätskonflikte B: Mittelbar gestaltbare Aufgabenmerkmale (soziale Konfliktaustragung, Kooperation, Entscheidungen, Planen, Niveau kognitiver Anforderungen, Qualifikationsanspruchnahme)	Gestaltung industrieller Montage-, Bedien- und Überwachungstätigkeit: – Sicherung vollständiger Tätigkeitsinhalte – Vermeidung von Regulationsbeeinträchtigungen
VERA – Verfahren zur Ermittlung von Regulationserfordernissen in der Arbeitstätigkeit Volpert et al. (1983)	Persönlichkeitsförderlichkeit: Niveau der Handlungsregulation nach einem Zehn-Stufen-Modell psychischer Regulationserfordernisse	Gestaltung industriell-gewerblicher Arbeitstätigkeit: Schaffung von Arbeitsaufgaben mit hohen kognitiven Regulationserfordernissen

KABA – Kontrastive Aufgabenanalyse im Büro Dunckel et al. (1993)	Humankriterien: Entscheidungsspielraum, Kommunikation, Belastungen, Zeitspielraum, Variabilität, Kontakt, Körperliche Aktivität, Strukturierbarkeit	Gestaltung von EDV-gestützten Arbeitssystemen vor dem Hintergrund der Humankriterien
RHIA – Regulationshindernisse in der Arbeitstätigkeit Leitner, Volpert, Greiner, Weber und Hennes (1987)	Beeinträchtigungslosigkeit: Regulationsbehinderungen als A: Regulationshindernisse (Erschwerungen, Unterbrechungen) B: Regulationsüberforderungen (aufgabenimmanent, aufgabenspezifisch)	Gestaltung industriell-gewerblicher Arbeitstätigkeit: Abbau psychisch belastender Arbeitsaufgaben- und -bedingungsmerkmale
ISTA – Instrument zur stressbezogenen Arbeitsanalyse (Beobachtungsversion) Semmer (1984)	Beeinträchtigungslosigkeit: A: Stressoren (unangemessener Regulationsaufwand, Regulationsunsicherheit), Zielunsicherheit B: Allgemeine Regulationsanforderungen und Ressourcen	Vergleich von Belastungsstrukturen bei unterschiedlichen gewerblichen Arbeitstätigkeiten
TAI – Tätigkeitsanalyseinventar Frieling, Kannheiser, Facaoaru, Wöcherl und Dürholt (1984)	Modularer Aufbau mit vier Teilverfahren zu den Problemschwerpunkten: I Emotional beanspruchungsrelevante Tätigkeitsbedingungen. II Kognitiv beanspruchungsrelevante Tätigkeitsbedingungen (informatorische Belstungen) III Qualifikationsanfoderungen IV Erfolgte und zu erwartende Veränderungen	Universell einsetzbar (geistige und manuelle Arbeitstätigkeiten): Bildung und Vergleich von Bedingungs- und Anforderungskonfigurationen; Ableitung von Gestaltungs- und Qualifizierungserfordernissen; Evaluation von Veränderungsprozessen
b) Arbeitswissenschaftlich/verhaltenstheoretisch fundierte Verfahren		
FAA – Fragebogen zur Arbeitsanalyse Frieling und Hoyos (1978)	Angemessene Belastung/ Beanspruchung: - Informationsaufnahme und -verarbeitung - Arbeitsausführung (Arbeitsmittel, Bedienelemente, manuelle Tätigkeiten) - Umgebungseinflüsse und besondere Arbeitsbedingungen	Universell einsetzbar (geistige und manuelle Arbeitstätigkeiten): vergleichende Arbeits- und Berufsanalysen; Klassifikation von Arbeitstätigkeiten und Berufen; Ableitung von Eignungsvoraussetzungen; Entwicklung von Lern- und Trainingsprogrammen
AET – Arbeitswissenschaftliches Erhebungsverfahren zur Tätigkeitsanalyse Rohmert und Landau (1979)	Angemessene Belastung/Beanspruchung: Formen menschlicher Arbeit in Arbeitssystemen (energetisch-effektorische, informatorische Tätigkeitselemente); Arbeitsobjekte und Arbeitsmittel; Umgebungseinflüsse (Beleuchtung, Lärm, Vibration, Klima, Arbeitssicherheit); organisatorische und wirtschaftliche Aspekte	Universell einsetzbar (geistige und manuelle Arbeitstätigkeiten): – vergleichende Arbeitssystembeschreibungen – Ableitung von Anforderungen

2. Schriftliche Befragungen (Fragebogen)		
JDS -– Job Diagnostic Survey Hackman und Oldham (1975)	Indices für das 'Motivating Potential Score (MPS)': Anforderungsvielfalt, Ganzheitlichkeit der Aufgabe, Bedeutsamkeit der Aufgabe für andere, Autonomie, Rückmeldung	Universell einsetzbar (geistige und manuelle Arbeitstätigkeiten): Vergleich von motivationspsychologisch relevanten Tätigkeitsmerkmalen
SAA -–Fragebogen zur subjektiven Arbeitsanalyse Udris und Alioth (1980)	A Handlungsspielraum B Transparenz C Verantwortung D Qualifikation E Soziale Struktur F Arbeitsbelastung	Universell einsetzbar (geistige und manuelle Arbeitstätigkeiten): Vergleich von Belastungsstrukturen in der Arbeitstätigkeit
ISTA -– Instrument zur stressbezogenen Arbeitsanalyse (Fragebogenversion) Semmer (1984)	siehe ISTA (Beobachtungsversion)	siehe ISTA (Beobachtungsversion)

Siehe für weitergehende Informationen und vertiefte Darstellungen:

Baitsch C., Katz C., Spinas P. & Ulich E. (1989) Computerunterstützte Büroarbeit. Zürich: Verlag der Fachvereine.

Duell W. & Frei F. (1986; Hrsg.) Arbeit gestalten – Mitarbeiter beteiligen. Frankfurt: Campus.

Duell W. & Frei F. (1986) Leitfaden für qualifizierende Arbeitsgestaltung. Köln: TÜV Rheinland.

Duell W. & Katz C. (1990) Ratgeber Bildschirmarbeit. (Schriftenreihe der Bundesanstalt für Arbeitsschutz, Forschungsanwendung FA 24). Bremerhaven: Verlag für neue Wissenschaften.

Koch M., Reiterer H. & Gärtner J. (1990) EDV im Büro – Handbuch zur menschengerechten Gestaltung. Wien: Oldenbourg.

Frieling E. & Sonntag K. (1987) Lehrbuch Arbeitspsychologie. Bern: Huber.

Hacker W. (1986) Arbeitspsychologie. Bern: Huber.

Ulich E. (1991, 3. Auflage 1994) Arbeitspsychologie. Zürich: Verlag der Fachvereine, Stuttgart: Poeschel-Verlag.

Literatur

AMI (1992) A quantitative approach to software management. Centre for Systems and Software Engineering. South Bank University. 103 Borough Road, London SE1 0AA (UK).

ANDRIOLE S. (1989) Storyboard prototyping – a new approach to user requirements analysis. Wellesley: QED Information Sciences.

AUGUST J. H. (1991) Joint Application Design: the Group Session Approach to system design. London: Prentice Hall.

BAITSCH C., KATZ C., SPINAS P. & ULICH E. (1989) Computerunterstützte Büroarbeit. Zürich: Verlag der Fachvereine.

BALZERT H. (1986) Die Entwicklung von Software-Systemen: Prinzipien, Methoden, Sprachen, Werkzeuge. (Reihe Informatik, Bd. 34). Zürich: Bibliographisches Institut.

BALZERT H., HOPPE H. U., OPPERMANN R., PESCHKE H., ROHR G. & STREITZ N. A. (1988; Hrsg.) Einführung in die Software-Ergonomie. (Reihe Mensch Computer Kommunikation – Grundwissen 1; H. Balzert, Hrsg.). Berlin: de Gruyter.

BARTSCH-SPÖRL B. (1991) Erfahrungen aus Projekten, die nicht im Lehrbuch stehen. In: P. Elzer (Hrsg.) Multidimensionales Software-Projektmanagement. (S. 291–314). Hallbergmoos: AIT.

BECK A. & ZIEGLER J. (1991) Methoden und Werkzeuge für die frühen Phasen der Software-Entwicklung. In: D. Ackermann & E. Ulich (Hrsg.) Software-Ergonomie'91. (Berichte des German Chapter of the ACM, Band 33, S. 76–85). Stuttgart: Teubner.

BENZ C. & HAUBNER P. (1983) Gestaltung von Bildschirmmasken. *Office Management* 31, S. 36–39.

BOEDICKER D. (1990) Handbuch-Knigge – Software-Handbücher schreiben und beurteilen. (Angewandte Informatik, Band 6; H. Balzert, Hrsg.). Mannheim: BI Wissenschaftsverlag.

BOEHM B. (1981) Software Engineering Economics. Englewood Cliffs: Prentice Hall.

BOOCH G. (1994; 2nd ed.) Object-oriented analysis and design with applications. Redwood City: The Benjamin/Cummings Publ.

BORTZ J. (1984) Lehrbuch der empirischen Sozialforschung. Berlin: Springer.

BROWN C. M. (1989) Human-Computer Interface Design Guidelines. Norwood: Ablex.

CHIKOFSKY E. J. & RUBENSTEIN B. L. (1988) CASE: Reliability engineering for information systems. *IEEE Software* 5 (2), S. 11–17.

DEMARCO T. & LISTER T. (1991) Wien wartet auf dich! Der Faktor Mensch im DV-Management. München: Hanser.

DEMING W. E. (1989) Out of the crisis. Cambridge, Mass.: Massachusetts Institute of Technology

DIAPER D. (1989; ed.) Task Analysis for Human-Computer Interaction. New York: Ellis Horwood.

DIX A. J. (1991) Formal Methods for Interactive Systems. London: Academic Press.

DUELL W. & FREI F. (1986) Leitfaden für qualifizierende Arbeitsgestaltung. Köln: TÜV Rheinland.

DUELL W. & KATZ C. (1990) Ratgeber Bildschirmarbeit. (Schriftenreihe der Bundesanstalt für Arbeitsschutz, Forschungsanwendung FA 24). Bremerhaven: Verlag für neue Wissenschaften.

DUMAS J. S. & REDISH J. C. (1993) A practical guide to usability testing. Norwood: Ablex.

DUNCKEL H., VOLPERT W., ZÖLCH M., KREUTNER U., PLEISS C. & HENNES K. (1993) Das KABA-Verfahren. (Schriftenreihe Mensch-Technik-Organisation, Band 5; E. Ulich, Hrsg.). Stuttgart: Teubner.

DUTKE S. (1994) Mentale Modelle: Konstrukte des Wissens und Verstehens. (Arbeit und Technik, Band 4; M. Frese & H. Oberquelle, Hrsg.). Göttingen: Hogrefe.

EHN P. & KYNG M. (1991) Cardboard Computers: Mocking-it-up or Hands-on the Future. In: J. Greenbaum & M. Kyng (eds.) Design at Work: Cooperative Design of Computer Systems. (S. 169–195). Hillsdale: Lawrence Erlbaum.

ERICSSON K. A. & SIMON H. A. (1984) Protocol Analysis. Cambridge: Cambridge University Press.

FLOYD C. (1984) A systematic look at prototyping. In: R. Budde, K. Kuhlenkamp, L. Mathiassen & H. Züllighoven (Hrsg.) Approaches to Prototyping. (S. 1–18). Berlin: Springer.

FREEDMAN D. P. & WEINBERG G. M. (1990) Handbook of Walkthroughs, Inspections, and Technical Reviews. New York: Dorset House Publ.

FREI F. & UDRIS I. (1990; Hrsg.) Das Bild der Arbeit. Bern: Huber.

FRESE M. & BRODBECK F. (1989) Computer in Büro und Verwaltung. Berlin: Springer.

FRESE M., FRITZ A., STOLTE W. & BRODBECK F. (1991) Eine erste Analyse von Tätigkeiten und ihre Anforderungen. Posterbeitrag auf der Arbeitstagung "Software für die Arbeit von morgen". München.

FRIELING E. & HOYOS C. (1978) Fragebogen zur Arbeitsanalyse (FAA). Bern: Huber.

FRIELING E., KANNHEISER W., FACAOARU C., WÖCHERL H. & DÜRHOLT E. (1984) Entwicklung eines theoriegeleiteten, standardisierten, verhaltenswissenschaftlichen Verfahrens zur Tätigkeitsanalyse (TAI). (Institut für Psychologie). München: Universität.

FROSCHAUER U. & LUEGER M. (1992) Das qualitative Interview zur Analyse sozialer Systeme. Wien: Universitätsverlag der Hochschülerschaft Uni Wien.

FRÜHAUF K., LUDEWIG J. & SANDMAYR H. (1991a) Software-Projektmanagement und -Qualitätssicherung. Zürich: Verlag der Fachvereine.

FRÜHAUF K., LUDEWIG J. & SANDMAYR H. (1991b) Software-Prüfung – eine Fibel. Zürich: Verlag der Fachvereine.

GALITZ W. (1989) Handbook of Screen Format Design. P.O. Box 181, Wellesley, MA 02181 (USA).

GEMÜNDEN G. (1990) Erfolgsfaktoren des Projektmanagements - eine kritische Bestandsaufnahme der empirischen Untersuchungen. *Projekt Management* 1&2, S. 4–15.

GNIECH G. (1976) Störeffekte in psychologischen Experimenten. Stuttgart: Kohlhammer.

GREENBAUM J. & KYNG M. (1991, eds.) Design at Work: Cooperative Design of Computer Systems. Hillsdale: Lawrence Erlbaum.

HABERFELLNER R., NAGEL P., BECKER M., BÜCHEL A., & VON MASSOW H. (1992) Systems Engineering – Methodik und Praxis. Zürich: Industrielle Organisation.

HACKER W. (1989) Designing the designer's Tasks: Participative Analysis and Evaluation of Software Development Tasks. In: M. Smith & G. Salvendy (eds.) Work with Computers: Organizational, Management, Stress and Health Aspects. (p. 163–168). Elsevier: Amsterdam.

HACKER W., IWANOWA A. & RICHTER P. (1983) Tätigkeitsbewertungssystem TBS. Berlin: Humboldt Uni, Psychodiagnostisches Zentrum.

HACKMAN J. & OLDHAM G. (1975) Development of the job diagnostic survey. *Journal of Applied Psychology* 60, S. 159–170.

HAMPE-NETELER W. & RÖDIGER K-H. (1991) Software-Ergonomie: Verfahren der Evaluierung und Standards zur Entwicklung von Benutzungsoberflächen. (Fachbereich Mathematik/Informatik, Bericht Nr. 2/92). Bremen: Universität Bremen.

HEILMANN H. (1981). Modelle und Methoden der Benutzermitwirkung in Mensch-Computer-Systemen. Stuttgart: Forkel.

HELANDER M. (1991; ed.) Handbook of Human-Computer Interaction. Amsterdam: North-Holland.

HESSE W., MERBETH G. & FRÖLICH R. (1992) Software-Entwicklung. (Handbuch der Informatik, Band 5.3). München: Oldenbourg.

HIX D. & SCHULMAN R. S. (1991) Human-Computer Interface Development Tools: a methodology for their evaluation. *Communications of the ACM* 34(3), p.75–87.

HOFFMANN T., KLOSE H.-G. & MARTIN H. (1989) Handbuch zur softwareergonomischen Gestaltung von Bildschirmmasken. (VDI Fortschrittsberichte Reihe 10: Informatik/Kommunikationstechnik Nr. 103) Düsseldorf: VDI Verlag.

JAMES B. (1989) The Systems Analysis Interview. Manchester: NCC Blackwell.

KIEBACK A., LICHTER H., SCHNEIDER-HUFSCHMIDT M. & ZÜLLIGHOVEN H. (1991) Prototyping in industriellen Software-Projekten – Er-

fahrungen und Analysen. (GMD-Studie Nr. 184). Sankt Augustin: GMD.

KISSLER L. (1988; Hrsg.) Computer und Beteiligung. Beiträge aus der empirischen Partizipationsforschung. Opladen: Westdeutscher Verlag.

KLOTZ U. (1991) Die zweite Ära der Informationstechnik. *Harvard Manager* 13(2), S. 101–112.

KOCH M., REITERER H. & GÄRTNER J. (1990) EDV im Büro – Handbuch zur menschengerechten Gestaltung. Wien: Oldenbourg.

KOSLOWSKI K. (1988) Unterstützung von partizipativer Systementwicklung durch Methoden des Softwareengineering. Opladen: Westdeutscher Verlag.

KUBICEK H. (1979) Interessenberücksichtigung beim Technikeinsatz im Büro- und Verwaltungsbereich. (Bericht Nr. 125 der Gesellschaft für Mathematik und Datenverarbeitung). München: Oldenbourg.

LAUTER B. (1987) Software-Ergonomie in der Praxis. Wien: Oldenbourg.

LEITNER K., VOLPERT W., GREINER B., WEBER W. & HENNES K. (1987) Analyse psychischer Belastung in der Arbeit (RHIA). Köln: TÜV Rheinland.

LIKERT R. (1972) Neue Aspekte der Unternehmensführung. (Schriftenreihe "Führung und Organisation der Unternehmung", Band 14). Bern: Paul Haupt.

MAASS S. (1994) Transparenz – eine zentrale Software-ergonomische Forderung. (Berichte des Fachbereiches Informatik FBI-HH-B-170/94). Hamburg: Universität Hamburg.

MAMBREY P. & OPPERMANN R. (1983; Hrsg.) Beteiligung von Betroffenen bei der Entwicklung von Informationssystemen. Frankfurt: Campus.

MAMBREY P., OPPERMANN R. & TEPPER A. (1986) Computer und Partizipation. Ergebnisse zu Gestaltungs- und Handlungspotentialen. Opladen: Westdeutscher Verlag.

MARTIN C. F. (1988) User-Centered Requirements Analysis. Englewood Cliffs: Prentice Hall.

MARTIN J. & MCCLURE C. (1985) Diagramming Techniques for Analysts and Programmers. Englewood Cliffs: Prentice-Hall.

MAUCH H. (1981) Metaplan. Nitor-Kommunikationstechnik GmbH, Jacksteinweg 10a, D-2000 Hamburg 52, Telefon +49-40-826 363.

MAYHEW D. J. (1992) Principles and Guidelines in Software User Interface Design. Englewood Cliffs: Prentice Hall.

MEYER-SCHÖNHERR M. (1992) Szenario-Technik. Ludwigsburg: Wissenschaft & Praxis.

MITTENECKER E. (1987) Video in der Psychologie. (Methoden der Psychologie, Band 9; K. Pawlik, Hrsg.). Bern: Huber.

MOLL T. (1987) Über Methoden zur Analyse und Evaluation interaktiver Computersysteme. In: K.-P. Fähnrich (Hrsg.) Software-Ergonomie. (State of the Art 5, German Chapter of the ACM, S. 179–190). München: Oldenbourg.

MULLER M. (1993) PICTIVE: Democratizing the Dynamics of the Design Session. In: D. Schuler & A. Namioka (eds.) Participatory Design: Principles and Practices. (S. 211–237). Hillsdale: Lawrence Erlbaum.

MÜLLER-HOLZ AUF DER HEIDE B., ASCHERSLEBEN G., HACKER S. & BARTSCH T. (1991) Methoden zur empirischen Bewertung der Benutzerfreundlichkeit von Bürosoftware im Rahmen von Prototyping. In: M. Frese, C. Kasten, C. Skarpelis & B. Zang-Scheucher (Hrsg.) Software für die Arbeit von morgen. (S. 409–420). Berlin: Springer.

MUMFORD E. & WELTER G. (1984) Benutzerbeteiligung bei der Entwicklung von Computersystemen. Berlin: Schmidt.

MUMMENDEY H.D. (1987) Die Fragebogen-Methode. Göttingen: Hogrefe.

MUSiC (1993) Metrics for Usability Standards in Computing. Teddington: National Physical Laboratory.

NIELSEN J. (1993) Usability Engineering. San Diego: Academic Press.

NIELSEN J. (1994, Ed.) Usability Laboratories. Special Issue of *Behaviour and Information Technology* 13(1–2).

NORMAN D. A. (1983) Design rules based on analyses of human error. *Communication of the ACM* 26(4), p. 254–258.

NORMAN D. A. (1989) Dinge des Alltags – Gutes Design und Psychologie für Gebrauchsgegenstände. Frankfurt: Campus.

OBERQUELLE H. (1987) Sprachkonzepte für Benutzergerechte Systeme. (Informatik-Fachberichte, Band 144). Berlin: Springer.

OPPERMANN R. (1983) Forschungsgegenstand und Perspektiven partizipativer Systementwicklung. München: Oldenbourg.

OPPERMANN R., MURCHNER B., REITERER H. & KOCH M. (1992; 2. Auflage) Software-ergonomische Evaluation: Der Leitfaden EVADIS II. Berlin: de Gruyter.

OTT H. J. (1991) Software-Systementwicklung: praxisorientierte Verfahren und Methoden. München: Hanser.

PESCHKE H. (1986) Betroffenenorientierte Systementwicklung. Frankfurt: Lang.

POMBERGER G. & BLASCHEK G. (1993) Grundlagen des Software-Engineering: Prototyping und objektorientierte Software-Entwicklung. München: Hanser.

RAASCH J. (1991) Systementwicklung mit strukturierten Methoden: ein Leitfaden für Praxis und Studium. München: Hanser.

RAUTERBERG M. (1991) Benutzungsorientierte Benchmarktests: eine Methode zur Benutzerbeteiligung bei der Entwicklung von Standardsoftware. (Projektberichte zum BOSS-Projekt Nr. 6; P. Spinas, M. Rauterberg, O. Strohm, D. Waeber & E. Ulich, Hrsg.). Zürich: ETH, Institut für Arbeitspsychologie.

RAUTERBERG M. (1993) AMME: an Automatic Mental Model Evaluation to analyze user behaviour traced in a finite, discrete state space. *Ergonomics* 36(11), p. 1369-1380.

RAUTERBERG M. & STROHM O. (1992) Work organization and software development. *Annual Review of Automatic Programming* 16(2), p. 121–128.

RICHARDSON S., DOHRENWEND B. & KLEIN D. (1965) Interviewing – Its Forms and Functions. New York: Basic Books.

RICHTER P., HEIMKE K. & MALESSA A. (1988) Tätigkeitspsychologische Bewertung und Gestaltung von Arbeitsaufgaben. *Zeitschrift für Arbeits- und Organisationspsychologie* 32, S. 13–21.

RÖDIGER K.-H., HAMPE-NETELER W. & PIEPENBURG U. (1991) Software-Ergonomie: Gestaltungsgrundsätze der DIN-Norm 66234 Teil 8 und ihre Umsetzung. Oberhausen: Technologie Beratungsstelle des DGB.

ROHMERT W. & LANDAU K. (1979) Das arbeitswissenschaftliche Erhebungsverfahren zur Tätigkeitsanalyse (AET). Bern: Huber.

RUCH L., SPINAS P. & ULICH E. (1989) Computerunterstützte Büroarbeit. Die Orientierung Nr. 95. Bern: Schweizerische Volksbank.

RUDOLPH E., SCHÖNFELDER E. & HACKER W. (1987) Tätigkeitsbewertungssystem – geistige Arbeit. Berlin: Humboldt Uni, Psychodiagnostisches Zentrum.

RUPIETTA W. (1987) Benutzerdokumentation für Softwareprodukte. (Angewandte Informatik, Band 3), Mannheim: B.I.

SCHIEMENZ B. (1979) Kybernetik. In: W. Kern (Hrsg.) Handwörterbuch der Produktionswissenschaft. (S. 1022–1028). Stuttgart: Poeschel.

SCHREIBER J. (1994) Beschaffung von Informatikmitteln: Pflichtenheft – Evaluation – Entscheidung. Bern: Schweizerische Vereinigung für Datenverarbeitung.

SCHÜPBACH H. (1993) Analyse und Bewertung von Arbeitstätigkeiten. In: H. Schuler (Hrsg.) Lehrbuch Organisationspsychologie. (S. 167–187). Bern: Huber.

SEMMER N. (1984). Stressbezogene Tätigkeitsanalyse. Weinheim: Beltz.

SOMMERVILLE I. (1989) Software Engineering. New York: Addison-Wesley.

SPINAS P., TROY N. & ULICH E. (1983) Leitfaden zur Einführung und Gestaltung von Arbeit mit Bildschirmsystemen. München: CW.

STEWART C. (1983) Interviewing: Principles and Practices, a Project Text. Dubuque: Kendall/Hunt.

STREITZ N. (1990) Psychologische Aspekte der Mensch-Computer-Interaktion. In: C. Hoyos & B. Zimolong (Hrsg.) Ingenieurspsychologie. (Enzyklopädie der Psychologie, Themenbereich D – Praxisgebiete, Serie III, Wirtschafts-, Organisations- und Arbeitspsychologie, Band 2, S. 240-284). Göttingen: Hogrefe.

STROHM O. & ULICH E. (1991) Arbeitsteilung und Benutzerorientierung bei der Software-Entwicklung. In: P. Elzer (Hrsg.) Multidimensionales Software-Projektmanagement.(S. 261–289). Hallbergmoos: AIT.

TROY N., BAITSCH C. & KATZ C. (1986) Bürocomputer – Chance für die Organisationsgestaltung? (Arbeitswelt, Band 3, A. Alioth, Hrsg.). Zürich: Verlag der Fachvereine.

UDRIS I. & ALIOTH A. (1980) Fragebogen zur subjektiven Tätigkeitsanalyse (SAA). In: E. Martin, I. Udris, U. Ackermann & K. Oegerli (Hrsg.) Monotonie in der Industrie. (S. 61–68). Bern: Huber.

ULICH E. (1985) Einige Anmerkungen zur Software-Psychologie. *Sysdata* 16(10), S. 53–58.

ULICH E. (1986a) Aspekte der Benutzerfreundlichkeit. In: W. Remmele & M. Sommer (Hrsg.) Arbeitsplätze von morgen. (Berichte des German Chapter of the ACM, Band 27, S. 102-121). Stuttgart: Teubner.

ULICH E. (1986b) Neuorientierung des Managements. In: W. Duell & F. Frei (Hrsg.) Arbeit gestalten – Mitarbeiter beteiligen. Eine Heuristik qualifizierender Arbeitsgestaltung. Frankfurt: Campus, S. 160–169.

ULICH E. (1989) Arbeitspsychologische Konzepte der Aufgabengestaltung. In: S. Maass & H. Oberquelle (Hrsg.) Software-Ergonomie'89. (Berichte des German Chapter of the ACM, Band 32, S. 51–65). Stuttgart: Teubner.

ULICH E. (1991) Gruppenarbeit – arbeitspsychologische Konzepte und Beispiele. In J. Friedrich & K.H. Rödiger (Hrsg.) Computergestützte Gruppenarbeit (CSCW). (Berichte des German Chapter of the ACM, Band 34, S. 57–77). Stuttgart: Teubner.

ULICH E. (1994; 3.Auflage) Arbeitspsychologie. Zürich: vdf Hochschulverlag AG.

VÖGELE S. (1992) Dialogmethode – das Verkaufsgespräch per Brief und Antwortkarte. Zürich: Moderne Industrie.

VOLPERT W. (1987) Psychische Regulation von Arbeitstätigkeiten. In: U. Kleinbeck & J. Rutenfranz (Hrsg.), Arbeitspsychologie. (Enzyklopädie der Psychologie, Themenbereich D, Serie III, Band 1, S. 1–42), Göttingen: Hogrefe

VOLPERT W., OESTERREICHER R., GABLENZ-KOLAKOVIC S., KROGOLL T. & RESCH M. (1983) Verfahren zur Ermittlung von Regulationserfordernissen in der Arbeitstätigkeit (VERA). Köln: TÜV Rheinland.

WALLMÜLLER E. (1990) Software-Qualitätssicherung in der Praxis. München: Hanser.

WANDKE H. (1988) Mensch-Rechner-Interaktion: Optimierung des Informationsaustauschs. *wissenschaft und fortschritt* 40(7), S. 173–177.

WANDMACHER J. (1993) Software-Ergonomie. (Mensch-Computer-Kommunikation: Grundwissen, Band 2), Berlin New York: de Gruyter.

WICKE W. (1988) Methoden der Partizipation bei der Entwicklung computergestützter Arbeitssysteme. In: L. Kissler (Hrsg.) Computer und Beteiligung. Beiträge aus der empirischen Partizipationsforschung. (S. 117–140). Opladen: Westdeutscher Verlag.

WOOD J. & SILVER D. (1989) Joint Application Design – How to Design Quality Systems in 40% Less Time. New York: Wiley & Sons.

ZAPF D. & FRESE M. (1989) Benutzerfehler im Kontext von Arbeitsaufgabe und Arbeitsorganisation. In: S. Maass & H. Oberquelle (Hrsg.) Software-Ergonomie'89. (Berichte des German Chapter of the ACM, Band 32, S. 213–222). Stuttgart: Teubner.

ZIEGLER J. & ILG J. (1993; Hrsg.) Benutzergerechte Software-Gestaltung. München: Oldenbourg.

ZÖLCH M. & DUNCKEL H. (1991) Erste Ergebnisse des Einsatzes der "Kontrastiven Aufgabenanalyse". In: D. Ackermann & E. Ulich (Hrsg.) Software-Ergonomie '91. (Berichte des German Chapter of the ACM, Band 33, S. 363–372). Stuttgart: Teubner.

Personenverzeichnis